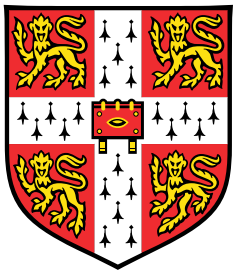
Towards Modeling the Geometry of Data Manifolds through Riemannian Geometry

A Framework for Representation Learning and Generative Modeling



Georgios Batzolis

Department of Applied Mathematics and Theoretical Physics
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Churchill College December 2024

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Georgios Batzolis
December 2024

Acknowledgements

And I would like to acknowledge ...

Abstract

This is where you write your abstract ...

Notation

This section introduces the notations used throughout the thesis, combining statistical concepts with elements from differential and Riemannian geometry. For details on Riemannian geometry, see [12, 15, 53, 77].

Statistical Notation

Random variables are denoted by capital letters (X, Y), with their associated probabilities written as $P(X), P(Y)$, and their densities as p_X, p_Y. Calligraphic letters $(\mathcal{X}, \mathcal{Y})$ represent sample spaces, while specific realizations of random variables are denoted by lowercase bold letters $(\mathbf{x}, \mathbf{y})$ for vectors and lowercase unbolded letters (x, y) for scalars. Given two random variables X and Y, the conditional probability of Y given X is denoted by $P(Y \mid X)$, and its density by $p_{Y|X}$. Unless explicitly stated otherwise, all probability measures are assumed to be absolutely continuous with respect to the Lebesgue measure, thereby admitting a density function.

Simplified Statistical Notation. In certain chapters, as noted at their outset, we adopt a streamlined statistical notation to improve clarity and simplify the presentation of formulas. For instance, probability distributions are denoted using only the argument of their density. For a random variable X_t with a realization x_t, we write:

$$p(x_t) := p_{X_t}(x_t).$$

Riemannian Geometry Notation

Let $\mathcal{M}$ denote a smooth manifold. The space of smooth functions on $\mathcal{M}$ is denoted by $C^\infty(\mathcal{M})$. The *tangent space* at $\mathbf{p} \in \mathcal{M}$, defined as the space of all *derivations* at $\mathbf{p}$, is written as $\mathcal{T}_\mathbf{p}\mathcal{M}$. Elements of $\mathcal{T}_\mathbf{p}\mathcal{M}$, called *tangent vectors*, are denoted by $\Xi_\mathbf{p} \in \mathcal{T}_\mathbf{p}\mathcal{M}$. The *tangent bundle* of $\mathcal{M}$ is $\mathcal{T}\mathcal{M}$, and smooth vector fields—smooth sections of the tangent bundle—are written as $\mathcal{X}(\mathcal{M}) \subset \mathcal{T}\mathcal{M}$.

A smooth manifold $\mathcal{M}$ becomes a *Riemannian manifold* if equipped with a smoothly varying *metric tensor field*, $(\cdot,\cdot) : \mathcal{X}(\mathcal{M}) \times \mathcal{X}(\mathcal{M}) \to C^\infty(\mathcal{M})$. This induces a *(Riemannian) metric*, $d_\mathcal{M} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$. The metric tensor also defines a unique affine connection, the *Levi-Civita connection*, denoted by $\nabla_{(\cdot)}(\cdot) : \mathcal{X}(\mathcal{M}) \times \mathcal{X}(\mathcal{M}) \to \mathcal{X}(\mathcal{M})$.

The Levi-Civita connection allows the definition of *geodesics*. A geodesic between two points $\mathbf{p}, \mathbf{q} \in \mathcal{M}$ is a curve $\gamma_{\mathbf{p},\mathbf{q}} : [0,1] \to \mathcal{M}$ of minimal length connecting $\mathbf{p}$ to $\mathbf{q}$. For

a tangent vector $\Xi_{\mathbf{p}} \in \mathscr{T}_{\mathbf{p}}\mathscr{M}$, the geodesic with initial velocity $\dot{\gamma}_{\mathbf{p},\Xi_{\mathbf{p}}}(0) = \Xi_{\mathbf{p}}$ defines the *exponential map*, $\exp_{\mathbf{p}} : \mathscr{D}_{\mathbf{p}} \to \mathscr{M}$, as:

$$\exp_{\mathbf{p}}(\Xi_{\mathbf{p}}) := \gamma_{\mathbf{p},\Xi_{\mathbf{p}}}(1),$$

where $\mathscr{D}_{\mathbf{p}} \subset \mathscr{T}_{\mathbf{p}}\mathscr{M}$ is the domain on which $\gamma_{\mathbf{p},\Xi_{\mathbf{p}}}(1)$ is well-defined. The *logarithmic map*, $\log_{\mathbf{p}} : \exp(\mathscr{D}'_{\mathbf{p}}) \to \mathscr{D}'_{\mathbf{p}}$, is the inverse of $\exp_{\mathbf{p}}$, valid on the domain where $\exp_{\mathbf{p}}$ is a diffeomorphism.

Finally, let $(\mathscr{M}, (\cdot,\cdot))$ be a d-dimensional Riemannian manifold, and $\mathscr{N}$ a d-dimensional smooth manifold. If $\phi : \mathscr{N} \to \mathscr{M}$ is a diffeomorphism, the *pullback metric* on $\mathscr{N}$ is defined as:

$$(\Xi, \Phi)^{\phi} := (D_{(\cdot)}\phi[\Xi_{(\cdot)}], D_{(\cdot)}\phi[\Phi_{(\cdot)}])_{\phi(\cdot)}, \tag{1}$$

where $D_{\mathbf{p}}\phi : \mathscr{T}_{\mathbf{p}}\mathscr{N} \to \mathscr{T}_{\phi(\mathbf{p})}\mathscr{M}$ is the differential of ϕ. This pullback metric equips $\mathscr{N}$ with a Riemannian structure, $(\mathscr{N}, (\cdot,\cdot)^{\phi})$, preserving geometric properties from $(\mathscr{M}, (\cdot,\cdot))$. In this thesis, pullback mappings follow 1, denoted with ϕ as a superscript, e.g., $d^{\phi}_{\mathscr{N}}(\mathbf{p},\mathbf{q})$, $\gamma^{\phi}_{\mathbf{p},\mathbf{q}}$, $\exp^{\phi}_{\mathbf{p}}(\Xi_{\mathbf{p}})$, and $\log^{\phi}_{\mathbf{p}} \mathbf{q}$.

Table of contents

List of figures **xv**

List of tables **xvii**

1 Introduction **1**
1.1 Generative Modeling 2
1.1.1 The Problem of Generative Modeling 2
1.1.2 Why Deep Learning for Generative Modeling? 3
1.1.3 Generative Adversarial Networks (GANs) 4
1.1.4 Variational Autoencoders (VAEs) 5
1.1.5 Normalizing Flows 10
1.1.6 Auto-Regressive Models 12
1.1.7 Energy-Based Models (EBMs) 14
1.1.8 Diffusion Models 16
1.2 Self-Supervised Learning 24
1.2.1 Motivation 24
1.2.2 Mathematical Formulation of SSL 25
1.2.3 Generative vs. Discriminative SSL 26
1.2.4 Utilizing Learned Representations in Downstream Tasks 33
1.2.5 Challenges and Limitations in Current Methodologies 35

2 Thesis Outline and Contributions **37**
2.1 CAFLOW: Conditional Autoregressive Flows 37
2.2 Non-Uniform Diffusion Models 39
2.3 Diffusion Models Encode the Intrinsic Dimension of Data Manifolds 41
2.4 Variational Diffusion Auto-encoder: Latent Space Extraction from Pre-Trained Diffusion Models 43
2.5 Score-Based Pullback Riemannian Geometry 44

References **47**

List of figures

2.1 From left to right: ideal dependencies in the i^{th} autoregressive component. Dual-Glow modeling assumption [92]; information is exchanged only between latent spaces having the same dimension. Our modeling assumption; we retain the dependencies between L_i and the latent spaces of lower dimension. 37

2.2 Left: unconditional normalizing flow architecture used to encode conditioning and conditioned images, denoted by $Y_n = Y$ and $W_n = W$ respectively, into a sequence of hierarchical latent variables. Right: design of the conditional transformation G_i^θ that models the i^{th} autoregressive component. The index of the flow i is omitted in both the transformed latent variable Z_j and the intermediate latent variables Z'_j for simplicity. 38

2.3 Illustration of a multi-scale diffusion model with three scales. Inspired by multi-scale normalizing flows, this approach diffuses different parts of the image tensor (transformed into multi-level Haar coefficients) at varying speeds. High-frequency detail coefficients diffuse progressively faster, with d_1 diffusing faster than d_2, d_2 faster than d_3, and so on, ensuring that all coefficients reach the same (very low) signal-to-noise ratio (SNR) at their respective terminal diffusion times: $t = 0.25, 0.5, 0.75$, and 1.0, respectively. The low-frequency approximation coefficients a_3 diffuse the slowest, completing their diffusion at $t = 1.0$. The multi-scale structure reduces the dimensionality of the diffusing tensor at each scale, enabling faster computation. Separate neural networks S_1, S_2, S_3, S_4 approximate the score functions at different intervals, leveraging the reduced dimensionality of the intermediate distributions. This hierarchical design mirrors the structure of multi-scale normalizing flows, improving training and sampling efficiency while maintaining high image generation quality. 39

2.4 Results from our conditional multi-speed diffusive estimator. 40

2.5 (Left) Visualization of the score field near the data manifold. (Right) Visualisation of the estimation of the manifold dimension using the trained diffusion model. 41
2.6 Comparison of original and reconstructed images on the FFHQ dataset using our ScoreVAE framework. The left panel presents the original images from the FFHQ dataset, while the right panel displays the corresponding reconstructions generated by ScoreVAE. The results highlight the effectiveness of ScoreVAE in capturing intricate details and preserving high fidelity, overcoming the limitations of traditional VAE models. 43
2.7 Approximate data manifolds learned by the Riemannian autoencoder generated by score-based pullback Riemannian geometry for three datasets. The orange surfaces represent the manifolds learned by the model, while the blue points correspond to the training data. Each manifold provides a convincing low-dimensional representation of the data, isometric to its respective latent space. 45

List of tables

Chapter 1

Introduction

Over the past decade, artificial intelligence (AI) has undergone a transformative evolution, driven by groundbreaking advancements in generative modeling and self-supervised learning (SSL). These two fields have redefined our ability to model complex data distributions and leverage vast amounts of unlabeled data, leading to significant breakthroughs in areas such as media synthesis [44, 69, 26, 47], multimodal learning [67, 100, 97], and scientific discovery [43, 90, 73, 102].

Generative modeling focuses on learning the underlying probability distributions of data to create realistic and diverse samples. Innovations such as Generative Adversarial Networks (GANs) [31], diffusion models [39, 87], and normalizing flows [71, 64] have set new benchmarks in quality and applicability, enabling realistic image generation [44], lifelike video creation [26], natural-sounding audio synthesis [47], and text-to-image generation [69]. Beyond creative media, these models play a critical role in scientific applications, such as protein structure prediction [43, 4], AI-assisted drug discovery [90, 28], and materials science [14], where they accelerate research by generating novel molecules and exploring chemical spaces. Furthermore, they address challenges in inverse problems, including super-resolution [76, 98] and inpainting [93, 55].

Self-supervised learning, on the other hand, has emerged as a powerful approach to harness the vast reservoirs of unlabeled data by uncovering its inherent structure. SSL enables models to learn meaningful representations that are crucial for downstream tasks, often serving as the backbone of state-of-the-art systems across diverse domains. These representations can be extracted using generative methods, which model the data distribution, or discriminative methods, such as contrastive learning [18], which focus on distinguishing between similar and dissimilar data samples.

Self-supervised learning has had a profound impact on foundational technologies. In natural language processing, it powers groundbreaking models like BERT [21] and GPT

[13], which underpin conversational AI and advanced language understanding systems. In computer vision, SSL reduces dependence on labeled data while delivering robust, scalable visual representations, driving improvements in tasks like object recognition and image segmentation. These advancements underscore SSL's versatility and its pivotal role in advancing AI across domains and modalities. Additionally, SSL has been transformative in multimodal learning, enabling seamless alignment and integration across modalities such as text, images, and audio. This capability has unlocked advanced tasks like generating descriptive image captions [41], synchronizing audio with video [2], retrieving relevant content across formats [91], and enhancing representation learning for multimodal datasets [67].

In this thesis, we focus on the theoretical and computational aspects of generative modeling and generative self-supervised learning. To provide the necessary background, the remainder of this introduction is divided into two sections: an overview of generative modeling (1.1) and an overview of self-supervised learning (1.2).

1.1 Generative Modeling

1.1.1 The Problem of Generative Modeling

Generative modeling sits at the forefront of machine learning, enabling us to capture the underlying structure of complex data and generate new, realistic samples. Whether it's creating lifelike images, composing music, or designing novel molecules, generative models aim to understand and replicate the distribution of data they're trained on.

Formally, let's consider a dataset $\mathscr{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^d$, comprising N independent samples drawn from an unknown data-generating distribution $P(X)$, which we refer to as the *target distribution*. The central goal of generative modeling is to create a model that approximates $P(X)$ as closely as possible based solely on these samples.

To learn the distribution $P(X)$, we seek a computationally tractable approximation $\hat{P}(X)$ based on the dataset $\mathscr{D}$ and our assumptions about $P(X)$. We measure how close $\hat{P}(X)$ is to $P(X)$ using a divergence $D(\hat{P}(X), P(X))$, such as the Kullback-Leibler divergence or the Wasserstein distance. Our goal is to minimize this divergence:

$$P^*(X) = \arg\min_{\hat{P}(X)} D(\hat{P}(X), P(X)).$$

Typically, we parameterize $\hat{P}(X)$ with a finite-dimensional vector $\theta \in \mathbb{R}^p$ for some $p > 0$, forming a family of models $P_\theta(X)$ and transforming the problem into optimizing over parameters:

$$\theta^* = \arg\min_\theta D(P_\theta(X), P(X)).$$

For practicality, $P_\theta(X)$ must be computationally tractable; we should be able to efficiently sample from it or evaluate its probability density function. Different generative modeling approaches balance tractability and fidelity to $P(X)$, leading to various methods suitable for different applications.

By tackling this problem, we aim not only to reproduce the data we observe but also to uncover the underlying mechanisms that generate it. This deep understanding empowers us to create models that can generate new, unseen data that is indistinguishable from real-world samples, thereby expanding the horizons of what machines can learn and create.

1.1.2 Why Deep Learning for Generative Modeling?

Non-parametric methods such as Kernel Density Estimation (KDE) have been widely used for modeling probability distributions in low-dimensional settings due to their flexibility in estimating the underlying density without assuming a specific parametric form [82]. However, scaling KDE to high-dimensional data is problematic. The sample complexity of KDE scales exponentially with the number of feature dimensions d, making it impractical for high-dimensional tasks [99]. Specifically, as per [81], the number of samples n required to achieve a target estimation error ε must satisfy the following condition:

$$n \geq \mathcal{O}\left(\varepsilon^{-\frac{4+d}{4}}\right).$$

This exponential dependence on d implies that even for moderate dimensions, the required sample size becomes infeasibly large, rendering KDE and similar methods ineffective for generative modeling in high dimensions.

In many real-world applications, however, data does not uniformly occupy the high-dimensional ambient space but is concentrated near a lower-dimensional manifold—a concept known as the *manifold hypothesis* [27]. For example, natural images, although high-dimensional when represented by pixel values, often lie on manifolds of much lower intrinsic dimensionality due to dependencies and correlations among pixels [75]. This implies that the effective complexity of the data is determined by the manifold's dimension k, where $k \ll d$.

While the manifold hypothesis suggests a reduction in effective dimensionality, leveraging this property requires models that can explicitly capture the underlying manifold structure.

Non-parametric methods like KDE do not inherently account for the manifold and treat each dimension independently, failing to exploit the dependencies among features [94]. As a result, they still suffer from high sample complexity despite the data's lower intrinsic dimensionality.

Parametric models with appropriate inductive biases can overcome this limitation by incorporating assumptions about the data's structure into the learning process. Deep learning architectures are particularly well-suited for this task due to their ability to integrate domain-specific inductive biases through their design [50]. For instance, Convolutional Neural Networks (CNNs) employ local connectivity and weight sharing to capture spatial hierarchies and local patterns in image data [51]. Similarly, Recurrent Neural Networks (RNNs) are designed to model temporal dependencies in sequential data [40].

By aligning model architecture with the intrinsic properties of the data manifold, deep learning models effectively reduce the hypothesis space to functions consistent with the data's structure, enhancing learning efficiency and reducing the required number of samples [66]. This structural approach explains why deep learning–based methods have become indispensable for generative modeling in high dimensions [30, 46].

In the past decade, deep learning has introduced flexible architectures that can effectively learn complex data distributions. This chapter reviews five main types of generative models: Generative Adversarial Networks (GANs), Variational Auto-encoders (VAEs), Normalizing Flows, Auto-Regressive Models, Energy-Based Models, and Diffusion Models, each with unique strengths and limitations.

1.1.3 Generative Adversarial Networks (GANs)

Introduced by Goodfellow et al. [31], Generative Adversarial Networks (GANs) consist of two neural networks: a *generator* G and a *discriminator* D, trained simultaneously in a minimax game. The generator maps samples $\mathbf{z}$ from a simple prior distribution $P(Z)$ (e.g., a standard normal distribution) to the data space $\mathbb{R}^d$, producing synthetic data $G(\mathbf{z})$. The discriminator outputs a probability $D(X) \in [0,1]$ indicating whether a sample X is real (from the training data) or generated (from G).

TRAINING OBJECTIVE. The training objective is formulated as a two-player minimax game with the value function $V(D,G)$:

$$\min_G \max_D V(D,G) = \mathbb{E}_{X\sim P(X)}[\log D(X)] + \mathbb{E}_{Z\sim P(Z)}[\log(1-D(G(Z)))]$$

where $P(X)$ is the target distribution represented by the training set $\mathscr{D}$.

GANs are trained through an adversarial process where the generator and discriminator are updated iteratively to improve against each other. The discriminator D learns to better distinguish real data from generated data, while the generator G aims to produce data that can fool D. This training continues until the generator produces samples indistinguishable from real data ($P_G(X) = P(X)$).

There are various formulations of the value function designed to enhance training stability and performance. For instance, the Wasserstein GAN (WGAN) [3] modifies the original objective using the Earth Mover's distance to address issues like mode collapse and improve convergence.

STRENGTHS. GANs have several notable strengths. One of their primary advantages is their ability to produce **high-quality samples** that are visually realistic and sharp, especially in image generation tasks. Additionally, GANs offer **efficient sampling**; once the generator is trained, it can rapidly produce new samples through a single forward pass, without the need for iterative sampling procedures common in other generative models.

WEAKNESSES. Despite their strengths, GANs also present significant weaknesses. A major challenge is the **training instability** inherent in their adversarial setup, as highlighted by Salimans et al. [78]. This can lead to issues such as mode collapse—where the generator produces limited varieties of samples—vanishing gradients, and oscillatory behavior during training. Moreover, GANs **do not provide an explicit density** function or likelihood estimate for the generated data, limiting their applicability in tasks that require probabilistic interpretations or quantitative assessments of sample likelihood.

1.1.4 Variational Autoencoders (VAEs)

Variational Autoencoders (VAEs), introduced by Kingma and Welling [46] and Rezende et al. [72], are a powerful class of latent variable models. They aim to simultaneously learn a compact, low-dimensional latent representation of data and a generative process capable of synthesizing new samples. This combination makes VAEs a cornerstone of representation learning and probabilistic generative modeling.

VAEs rely on the assumption that observed data $X \in \mathscr{X}$ is generated from unobserved latent variables $Z \in \mathscr{Z}$ through a probabilistic generative process. This process is characterized by two key components. The latent variable Z is assumed to follow a simple prior distribution, typically a standard normal distribution:

$$p_Z(\mathbf{z}) = \mathscr{N}(\mathbf{z}; \mathbf{0}, \mathbf{I}),$$

which encourages compact and smooth representations in the latent space. The observed variable X, conditioned on a latent sample $\mathbf{z}$, follows a parameterized conditional density:

$$p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(\mathbf{x}; \mu_\theta(\mathbf{z}), \sigma_\theta^2(\mathbf{z})\mathbf{I}),$$

where the mean $\mu_\theta(\mathbf{z}) \in \mathbb{R}^d$ and variance $\sigma_\theta^2(\mathbf{z}) \in \mathbb{R}$ are outputs of a neural network (decoder) with parameters θ. These assumptions define the generative model: latent variables are sampled from the prior, and data is generated conditionally through the decoder network.

The Challenge of Marginal Likelihood Optimization. The marginal log-likelihood of a data point $\mathbf{x}$ under the stated model assumptions is given by:

$$\log p_{X,\theta}(\mathbf{x}) = \log \int p_{X,Z,\theta}(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \log \int p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z})\, p_Z(\mathbf{z})\, d\mathbf{z}.$$

This integral is intractable because, for a given data point $\mathbf{x}$, only a small region in the latent space $\mathcal{Z}$ significantly contributes to the likelihood. This critical region corresponds to values of $\mathbf{z}$ where the conditional likelihood $p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z})$ is high, concentrating most of the reconstruction probability mass. Intuitively, this reflects the fact that each data point is typically explained by a specific combination of underlying generative factors, such as shape, texture, or orientation, which correspond to a small subset of the latent space. When attempting to estimate the integral using Monte Carlo sampling from the prior $p_Z(\mathbf{z})$, the vast majority of samples fall outside this region and contribute negligibly to the likelihood. This leads to highly inefficient sampling and high-variance estimates, as reliable integration would require an impractically large number of samples.

If we could sample directly from the true posterior $p_{Z|X,\theta}(\mathbf{z} \mid \mathbf{x})$, which focuses on this important region of the latent space, we could obtain reliable Monte Carlo estimates of the likelihood. However, the true posterior itself depends on the intractable marginal likelihood $p_{X,\theta}(\mathbf{x})$, making direct computation impractical. This is where the variational approximation of the posterior comes into play.

Variational Approximation of the True Posterior. To address this intractability, VAEs introduce a variational approximation $q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})$, modeled by a neural network (the encoder), to approximate the true posterior $p_{Z|X,\theta}(\mathbf{z} \mid \mathbf{x})$. The variational distribution $q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})$ is parameterized as a Gaussian:

$$q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mathbf{z}; \mu_\phi(\mathbf{x}), \mathrm{diag}(\sigma_\phi^2(\mathbf{x}))),$$

where $\mu_\phi(\mathbf{x}) \in \mathbb{R}^k$ and $\sigma_\phi(\mathbf{x}) \in \mathbb{R}^k$ are outputs of the encoder network with parameters ϕ and k denotes the dimension of the latent space.

The encoder is trained to match the true posterior as closely as possible, ensuring that the critical region of the latent space is sampled effectively. This enables low-variance Monte Carlo estimates of the marginal likelihood via importance sampling. The variational approximation thus transforms an otherwise intractable optimization problem into a practical one, paving the way for the derivation of the evidence lower bound (ELBO), a tractable objective that can be efficiently optimized to jointly train the encoder and decoder.

Derivation of the Evidence Lower Bound (ELBO). To address the intractability of directly optimizing the marginal log-likelihood $\log p_{X,\theta}(\mathbf{x})$, the variational approximation $q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})$ is utilized. This approximation allows efficient sampling from the critical region of the latent space that contributes most to the likelihood. Rewriting the marginal log-likelihood gives:

$$\log p_{X,\theta}(\mathbf{x}) = \log \int q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \frac{p_{X,Z,\theta}(\mathbf{x},\mathbf{z})}{q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})} d\mathbf{z}.$$

The term inside the integral represents an importance-weighted likelihood, where the variational distribution $q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})$ serves as the proposal distribution for efficiently sampling from the posterior.

Applying Jensen's inequality to the logarithm of the expectation yields:

$$\log p_{X,\theta}(\mathbf{x}) \geq \mathbb{E}_{q_{Z|X,\phi}(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_{X,Z,\theta}(\mathbf{x},\mathbf{z})}{q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})} \right].$$

The right-hand side is defined as the *Evidence Lower Bound* (ELBO), denoted as $\mathrm{ELBO}(\theta,\phi)$. By construction, the ELBO provides a tractable lower bound on the marginal log-likelihood and serves as the objective for training VAEs. Substituting the joint probability $p_{X,Z,\theta}(\mathbf{x},\mathbf{z}) = p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z})\, p_Z(\mathbf{z})$, the ELBO can be expressed as:

$$\mathrm{ELBO}(\theta,\phi) = \mathbb{E}_{q_{Z|X,\phi}(\mathbf{z}|\mathbf{x})} \big[\log p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z}) \big] - D_{\mathrm{KL}} \big(q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \,\|\, p_Z(\mathbf{z}) \big).$$

The ELBO consists of two terms:

1. **Reconstruction Term:**

$$\mathbb{E}_{q_{Z|X,\phi}(\mathbf{z}|\mathbf{x})} \big[\log p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z}) \big],$$

   which encourages the generative model $p_{X|Z,\theta}$ (decoder) to reconstruct the observed data $\mathbf{x}$ accurately from the latent representation $\mathbf{z}$.

2. **KL Divergence Term:**

$$D_{\mathrm{KL}}(q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \,\|\, p_Z(\mathbf{z})),$$

which regularizes the approximate posterior $q_{Z|X,\phi}$, ensuring that the learned latent representations are close to the prior p_Z.

Together, these terms balance accurate data reconstruction and latent space regularization. Maximizing the ELBO ensures the latent space encodes meaningful, generalizable representations while enabling robust reconstruction through the decoder.

Connection to the Marginal Log-Likelihood. To further understand the role of the ELBO, the marginal log-likelihood can be decomposed as follows:

$$\log p_{X,\theta}(\mathbf{x}) = \text{ELBO}(\theta,\phi) + D_{\text{KL}}\big(q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \,\|\, p_{Z|X,\theta}(\mathbf{z} \mid \mathbf{x})\big).$$

The second term is the KL divergence between the approximate posterior $q_{Z|X,\phi}$ and the true posterior $p_{Z|X,\theta}$. This term is always non-negative, implying that $\text{ELBO}(\theta,\phi) \leq \log p_{X,\theta}(\mathbf{x})$. From this decomposition, it additionally becomes clear that maximizing the ELBO serves a dual purpose:

1. It increases the lower bound on the marginal log-likelihood, improving the model's ability to explain the data.

2. It minimizes the mismatch between the approximate posterior $q_{Z|X,\phi}$ and the true posterior $p_{Z|X,\theta}$, ensuring effective latent representation learning.

TRAINING OBJECTIVE. The training objective of Variational Autoencoders (VAEs) extends from the Evidence Lower Bound (ELBO), balancing reconstruction accuracy and latent space regularization. The goal is to maximize the ELBO or equivalently minimize its negative form:

$$\mathscr{L}_{\text{VAE}}(\theta,\phi) = -\mathbb{E}_{q_{Z|X,\phi}(\mathbf{z}|\mathbf{x})}\big[\log p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z})\big] + D_{\text{KL}}\Big(q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \,\Big\|\, p_Z(\mathbf{z})\Big).$$

The first term ensures accurate reconstruction, while the second term regularizes the latent space to stay close to the prior p_Z. Optimization is made efficient using stochastic gradient descent and the *reparameterization trick* [46].

To provide more control over the trade-off between reconstruction and regularization, the β-ELBO [37] introduces a weighting factor $\beta > 0$ for the KL term:

$$\mathscr{L}_{\beta\text{-VAE}}(\theta,\phi) = -\mathbb{E}_{q_{Z|X,\phi}(\mathbf{z}|\mathbf{x})}\big[\log p_{X|Z,\theta}(\mathbf{x} \mid \mathbf{z})\big] + \beta\, D_{\text{KL}}\Big(q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x}) \,\Big\|\, p_Z(\mathbf{z})\Big).$$

- $\beta = 1$: Reduces to the standard ELBO.

- $\beta < 1$: Prioritizes reconstruction by underweighting regularization.

- $\beta > 1$: Enforces stronger regularization, which can promote disentanglement but may reduce reconstruction quality.

The β-ELBO offers flexibility to adapt VAEs for diverse tasks, balancing generative performance with latent structure. Smaller β values are often preferred to prevent over-regularization and improve reconstruction quality.

SAMPLING AND DISTRIBUTION MISMATCH. Once trained, the model can generate new samples $\mathbf{x}_{\text{new}}$ by first drawing $\mathbf{z} \sim P(Z)$ (e.g., $\mathcal{N}(\mathbf{0}, \mathbf{I})$) and then sampling $\mathbf{x}_{\text{new}} \sim p_\theta(X \mid Z)$. However, in practice, the *aggregated posterior*

$$q_Z(\mathbf{z}) \;=\; \int q_{Z|X,\phi}(\mathbf{z} \mid \mathbf{x})\; p_X(\mathbf{x})\; d\mathbf{x}$$

often differs from $p_Z(\mathbf{z})$. This mismatch can result in poor-quality samples if we draw $\mathbf{z}$ directly from the prior distribution, because the decoder rarely "sees" latent codes that lie far outside the support of $q_Z(\mathbf{z})$.

VAEs IN COMBINATION WITH OTHER GENERATIVE MODELS. To alleviate this mismatch, VAEs are frequently combined with other frameworks that can accurately model the aggregated posterior density $q_Z(\mathbf{z})$. For instance, **latent diffusion models** [74] are employed to model the aggregated posterior distribution. Here, the VAE's encoder compresses high-dimensional data $\mathbf{x}$ into a latent representation $\mathbf{z}$, and a diffusion model is trained to model $q_Z(\mathbf{z})$. At sampling time, the diffusion model generates $\mathbf{z}$ in the compressed space, and the VAE's decoder then reconstructs a high-resolution sample in $\mathcal{X}$. This factorization into a *learned, lower-dimensional space* significantly reduces the computational cost of training and sampling while often improving sample fidelity.

STRENGTHS. VAEs are a powerful framework that seamlessly combines generative modeling and representation learning, excelling in both domains. They provide compact, semantically meaningful latent representations that enable tasks such as data interpolation, clustering, and feature extraction, making them particularly effective for analyzing and manipulating high-dimensional data. Simultaneously, their generative capabilities allow for synthesizing high-quality samples from the latent space, with the probabilistic framework naturally incorporating uncertainty—a valuable property for anomaly detection, Bayesian inference, and robust decision-making under uncertainty. Unlike adversarial models like GANs, VAEs rely on the ELBO objective, ensuring stable and reliable training without issues like mode collapse. This stability, coupled with their flexibility, enables VAEs to integrate seamlessly with advanced generative techniques such as latent diffusion models, enhancing sample fidelity and computational efficiency, especially in high-dimensional settings. These

strengths position VAEs as a versatile and foundational tool for a wide range of applications in modern machine learning.

WEAKNESSES. Despite their strengths, VAEs face notable challenges that limit their generative performance. A primary issue is the tendency to produce blurred reconstructions and samples, stemming from the Gaussian decoder assumption, which simplifies the conditional likelihood and constrains the model's ability to capture complex data structures. This limitation is addressed in Chapter **??**, where we introduce *ScoreVAE*, a novel adaptation of the VAE framework. By combining a diffusion-time-dependent encoder with an unconditional diffusion model and leveraging Bayes' rule for score functions, *ScoreVAE* bypasses the Gaussian assumption and provides a more expressive reconstruction distribution, significantly enhancing the model's ability to capture intricate high-dimensional structures. Additionally, the mismatch between the aggregated posterior and the prior, which often leads to poor-quality samples, necessitates a model that learns the aggregated posterior distribution, such as a latent diffusion model or a latent normalizing flow, to improve sample fidelity and better align the generative process with the learned latent space.

1.1.5 Normalizing Flows

Normalizing flows, introduced by Rezende and Mohamed [71], are a versatile class of generative models that provide a unified framework for both data generation and explicit likelihood estimation. These models transform a simple base distribution into a complex target distribution using a sequence of invertible and differentiable mappings, enabling exact computation of likelihoods. Normalizing flows are trained by directly maximizing the log-likelihood of the observed data, leveraging the change of variables formula to compute densities in closed form. This combination of flexibility, tractability, and theoretical soundness makes normalizing flows a powerful tool for modeling complex probability distributions.

Given a latent variable $Z \sim P(Z)$ in latent space $\mathscr{Z}$ and an invertible function $G^\theta : \mathscr{Z} \to \mathscr{X}$ parameterized by θ, the model defines the random variable $X = G^\theta(Z)$ in data space $\mathscr{X}$ to approximate the data distribution $P(X)$. The transformation G^θ maps samples from the base distribution to samples resembling the observed data.

To compute the probability density function (pdf) of X, the change of variables formula is employed. For an invertible and differentiable transformation G^θ, with inverse $F^\theta = (G^\theta)^{-1}$, the pdf of X is given by:

$$p_X^\theta(\mathbf{x}) = p_Z\left(F^\theta(\mathbf{x})\right)\left|\det\left(\frac{\partial F^\theta(\mathbf{x})}{\partial \mathbf{x}}\right)\right|,$$

where $\frac{\partial F^\theta(\mathbf{x})}{\partial \mathbf{x}}$ is the Jacobian matrix of F^θ at point $\mathbf{x}$, and det denotes the determinant. Taking the logarithm yields:

$$\log p_X^\theta(\mathbf{x}) = \log p_Z\left(F^\theta(\mathbf{x})\right) + \log\left|\det\left(\frac{\partial F^\theta(\mathbf{x})}{\partial \mathbf{x}}\right)\right|.$$

This formulation enables exact computation of the log-likelihood, which is crucial for effective training.

In practice, the transformation G^θ is constructed by composing a sequence of K invertible and differentiable transformations:

$$G^\theta = G_K \circ G_{K-1} \circ \cdots \circ G_1,$$

where each G_k is an invertible function. The inverses of these transformations are denoted as $F_k = G_k^{-1}$.
For a data point $\mathbf{x}$, intermediate variables are defined by recursively applying the inverses:

$$\mathbf{h}_0 = \mathbf{x}, \quad \mathbf{h}_k = F_k(\mathbf{h}_{k-1}), \quad \text{for } k = 1, \ldots, K.$$

The log-density of $\mathbf{x}$ can then be expressed as:

$$\log p_X^\theta(\mathbf{x}) = \log p_Z(\mathbf{h}_K) + \sum_{k=1}^{K} \log\left|\det\left(\frac{\partial F_k(\mathbf{h}_{k-1})}{\partial \mathbf{h}_{k-1}}\right)\right|.$$

A crucial aspect of designing normalizing flows is ensuring that the composing functions G_k are, typically, analytically invertible to allow for fast sampling. Additionally, the calculation of the determinant of the Jacobian must be tractable to enable efficient training via the change of variables formula. This is often achieved by designing the transformations so that the resulting Jacobian matrices have a lower triangular form or another structure that simplifies the computation of the determinant. By constructing G_k in this manner, normalizing flows facilitate efficient evaluation of both the forward and inverse transformations, as well as the log-determinant of the Jacobian, thereby enabling effective training and fast sampling from the model.

TRAINING OBJECTIVE. The parameters θ of the normalizing flow are optimized by maximizing the likelihood of the observed data $\{\mathbf{x}_i\}_{i=1}^N$, which is equivalent to minimizing the negative log-likelihood:

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} -\log p_X^\theta(\mathbf{x}_i).$$

By leveraging the exact computation of the log-likelihood, normalizing flows can be trained using standard gradient-based optimization methods, ensuring stable and efficient convergence.

STRENGTHS. Normalizing flows offer several advantages. A primary strength is their ability to perform **exact density computation**, allowing for tractable and exact likelihood evaluations. This facilitates stable training via maximum likelihood estimation [64]. Furthermore, they provide **efficient sampling**; once trained, new samples can be generated rapidly by transforming latent samples through G^θ. The analytical invertibility of G^θ and efficient computation of Jacobian determinants make both training and sampling computationally efficient.

WEAKNESSES. Despite their strengths, normalizing flows have limitations. The requirement for invertibility and tractable Jacobians imposes constraints on the choice of transformations, potentially leading to **limited expressivity** [64]. This can hinder the model's ability to capture complex data distributions, especially those with intricate structures. Additionally, normalizing flows can encounter **topological limitations and numerical stability issues** when modeling targets with complicated topologies. In such cases, the model may become numerically non-invertible to approximate the target distribution closely, leading to stability problems and reducing the effectiveness of the flow [11, 20].

Normalizing flows, while powerful, **lack a semantically meaningful lower-dimensional latent space**, unlike GANs and Variational Auto-Encoders (VAEs). GANs and VAEs are designed with a lower-dimensional latent space, explicitly chosen a priori, enabling their effective use in downstream tasks such as compression, image-text alignment, and editing. In contrast, the latent space of normalizing flows is inherently tied to the ambient space's dimensionality, limiting its direct interpretability. In Chapter **??**, we address this limitation by adapting normalizing flows to acquire an interpretable latent space. Our approach combines an anisotropic base distribution with isometry regularization, drawing on principles from pullback Riemannian geometry. This training paradigm constructs a latent space where semantic significance is encoded by the variances learned in the base distribution. Latent dimensions capturing meaningful semantic information are characterized by high variances, while those encoding noise exhibit very low variance, enabling a clear distinction between essential and extraneous information.

1.1.6 Auto-Regressive Models

Auto-regressive models are a class of generative models that factorize the joint probability density of high-dimensional data into a product of conditional densities, leveraging the chain rule of probability. Notable examples include PixelCNN and PixelRNN [61], WaveNet [60],

and more recently, Transformer-based models [96], which have been applied across various domains such as natural language processing, image generation, and audio synthesis.

Given a data vector $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ in $\mathbb{R}^d$, the joint density $p_X(\mathbf{x})$ is decomposed into a product of conditional densities:

$$p_X(\mathbf{x}) = p_{X_1}(x_1) \prod_{i=2}^{d} p_{X_i|X_{1:i-1}}(x_i \mid x_{1:i-1}),$$

where $p_{X_1}(x_1)$ represents the marginal probability of the first variable, $X_{1:i-1}$ denotes all preceding random variables in a predefined ordering, and $x_{1:i-1}$ represents their corresponding realizations. This factorization enables the model to capture complex dependencies between variables by modeling each variable conditioned on all previous ones.

Each conditional density $p_{X_i|X_{1:i-1}}(x_i \mid x_{1:i-1})$ is modeled using a neural network that takes $x_{1:i-1}$ as input and outputs the parameters of the density for x_i. For instance, in PixelCNN, convolutional neural networks with masked filters are employed to ensure that the prediction of a pixel x_i depends only on pixels $x_{1:i-1}$ that come before it in the chosen ordering.

The masking technique is critical; it zeroes out connections in the network to prevent information from "future" variables influencing the prediction of the current variable, thus maintaining the auto-regressive property.

TRAINING OBJECTIVE. The parameters θ of the auto-regressive model are optimized by maximizing the likelihood of the observed data $\{\mathbf{x}_n\}_{n=1}^N$, which is equivalent to minimizing the negative log-likelihood:

$$\begin{aligned}\theta^* &= \arg\min_\theta \sum_{n=1}^{N} -\log p_{X,\theta}(\mathbf{x}_n) \\ &= \arg\min_\theta \sum_{n=1}^{N} \left[-\log p_{X_1,\theta}(x_{n,1}) + \sum_{i=2}^{d} -\log p_{X_i|X_{1:i-1},\theta}(x_{n,i} \mid x_{n,1:i-1}) \right],\end{aligned}$$

where $x_{n,i}$ represents the i-th variable in the n-th observed data sample, and $x_{n,1:i-1}$ denotes its preceding realizations. The inclusion of $p_{X_1,\theta}(x_{n,1})$ accounts for the marginal probability of the first variable, ensuring the complete factorization of the joint likelihood.

SAMPLING PROCESS. Sampling from auto-regressive models is inherently sequential. To generate a new data sample $\mathbf{x}$, the model starts by sampling x_1 from $p_{X_1}(x_1)$. Then, for $i = 2$ to d, each x_i is sampled from $p_{X_i|X_{1:i-1}}(x_i \mid x_{1:i-1})$, using the previously sampled values. This process continues until all variables have been generated.

STRENGTHS. Auto-regressive models have several notable strengths. A primary advantage is their ability to perform **exact likelihood computation**, providing an exact calculation of the data likelihood without any approximation. This facilitates stable training via maxi-

mum likelihood estimation, ensuring that the model accurately captures the underlying data density. Additionally, by conditioning each variable on all previous ones, auto-regressive models are highly effective at **modeling complex dependencies** in sequential or structured data. This makes them particularly well-suited for tasks such as language modeling, image generation, and audio synthesis, where capturing intricate patterns and dependencies is crucial. Furthermore, auto-regressive models offer **flexibility in handling different data types**; they can model both discrete and continuous data by appropriately choosing the form of the conditional densities, such as using categorical densities for discrete variables or Gaussian densities for continuous ones.

WEAKNESSES. Despite these strengths, auto-regressive models also present significant weaknesses. A major limitation is the **slow sampling process** inherent in their sequential nature. Generating a single data sample requires sampling each variable one at a time, conditioned on the previously generated variables. This makes the sampling process computationally intensive and time-consuming, especially for high-dimensional data like high-resolution images or long sequences, where the number of variables d is large. Moreover, the sequential dependency of variables **limits parallelization during sampling**, as each variable depends on the outcome of the previous ones, preventing the use of parallel computation techniques to speed up sample generation. This can be particularly problematic when rapid generation of many samples is required.

1.1.7 Energy-Based Models (EBMs)

Energy-Based Models (EBMs) [52] define an unnormalized probability density over data using an energy function $E_\theta(\mathbf{x})$. The density of a data point $\mathbf{x} \in \mathscr{X}$ is given as:

$$p_\theta(\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}))}{Z_\theta},$$

where $Z_\theta = \int_{\mathscr{X}} \exp(-E_\theta(\mathbf{x}))\, d\mathbf{x}$ is the partition function, a normalizing constant that ensures $p_\theta(\mathbf{x})$ integrates to 1 over $\mathscr{X}$. However, computing Z_θ is typically intractable for high-dimensional data, posing challenges for direct optimization and sampling.

TRAINING OBJECTIVE. EBMs are trained to assign low energy values to samples drawn from the data distribution (positive samples) and high energy values to samples from the model distribution (negative samples). A commonly used training objective is based on contrastive divergence [38], which minimizes the expected energy of positive samples while maximizing the expected energy of negative samples:

$$\mathscr{L}(\theta) = \mathbb{E}_{X^+ \sim P(X)}[E_\theta(X^+)] - \mathbb{E}_{X^- \sim P_\theta(X)}[E_\theta(X^-)],$$

where X^+ and X^- are random variables associated with the data distribution $P(X)$ and the model distribution $P_\theta(X)$, respectively.

To obtain negative samples $\mathbf{x}^-$, a standard approach is to use *Markov Chain Monte Carlo (MCMC)* methods, with Langevin Dynamics [48, 58] being a widely adopted technique. Langevin Dynamics is a stochastic gradient-based method that iteratively updates a sample $\mathbf{x}_t$ to move toward regions of higher probability density under the unnormalized model density $p_\theta(\mathbf{x})$. The iterative update rule is given by:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \nabla_{\mathbf{x}} E_\theta(\mathbf{x}_t) + \sqrt{2\eta}\,\varepsilon_t,$$

where $\eta > 0$ is the step size, and $\varepsilon_t \sim \mathcal{N}(\mathbf{0}, I)$ represents Gaussian noise added to incorporate stochasticity into the updates. The term $-\nabla_{\mathbf{x}} E_\theta(\mathbf{x}_t)$ drives $\mathbf{x}_t$ toward regions of lower energy, while the stochastic noise ε_t prevents the chain from getting trapped in local minima.

This approach ensures that the generated negative samples $\mathbf{x}^-$ align with the model's current energy landscape. The addition of noise also maintains ergodicity, which is crucial for exploring the sample space. In the limit of small step sizes η and sufficient iterations, the chain converges to the target distribution $P_\theta(X)$.

SAMPLING PROCESS. Sampling from EBMs follows the same procedure as obtaining negative samples. Since direct sampling is infeasible due to the intractability of Z_θ, MCMC methods like Langevin Dynamics are used to approximate samples from the target distribution $P_\theta(X)$. This iterative process converges to high-density regions of $P_\theta(X)$, making it suitable for generation tasks. However, the need for multiple iterations results in a slow sampling process, particularly for high-dimensional data.

STRENGTHS. EBMs offer several notable strengths. One of their primary advantages is their **flexibility** in representing complex distributions without requiring explicit normalization, as they define an unnormalized probability distribution directly using an energy function [52]. This flexibility allows EBMs to focus on modeling the energy landscape of the data, making them highly adaptable to diverse data types and problem settings. Unlike other generative models, such as GANs or normalizing flows, which often require specific assumptions about the form of the probability distribution or rely on explicit normalization, EBMs can represent distributions of arbitrary complexity by appropriately designing the energy function.

Furthermore, EBMs exhibit strong **compositionality**, allowing seamless integration with other models and the incorporation of domain-specific knowledge such as constraints or priors [25, 33]. This property enables EBMs to handle applications requiring hybrid generative processes or strict adherence to domain rules. By encoding physical laws or chemical properties directly into the energy function, EBMs can generate outcomes that comply with these constraints, making them particularly effective in scientific modeling tasks like physical

simulations and molecular design [25]. In structured prediction tasks, EBMs can impose priors to ensure outputs follow specific patterns or relationships, such as spatial coherence in semantic segmentation or hierarchical structures in knowledge graph synthesis [33, 52]. This compositional and flexible nature allows EBMs to excel in applications requiring constrained generative modeling or the integration of domain knowledge, where conventional generative models may struggle.

WEAKNESSES. Despite their strengths, EBMs also face significant weaknesses. A major challenge is their **training difficulty**, primarily due to the reliance on MCMC methods for generating negative samples. MCMC sampling is computationally intensive, can suffer from slow convergence, and often requires careful tuning to ensure effective exploration of the energy landscape. Additionally, EBMs exhibit **slow sampling speed**, as generating a single sample requires running iterative Langevin Dynamics or similar methods, which involve multiple gradient evaluations. This can hinder the scalability of EBMs for tasks requiring large-scale generation.

1.1.8 Diffusion Models

Denoising Diffusion Probabilistic Models (DDPMs) [39] and score-based generative models (SGMs) [88] generate data by reversing a noise corruption process. These models consist of two complementary components: a forward process that gradually perturbs data into a simple noise-like prior distribution (e.g., $\mathcal{N}(0, \mathbf{I})$) and a reverse process that progressively removes the noise, converting samples from the prior distribution into realistic data samples.

In the literature, discrete-time models such as DDPMs are widely referred to as "diffusion models", a term reflecting their reliance on discrete diffusion processes to corrupt data over a finite number of steps and then reverse this transformation. Continuous-time frameworks like SGMs, on the other hand, are often described as "score-based generative models," as they emphasize the estimation of the score function—the gradient of the log probability density—and use it to define the reverse process via stochastic differential equations (SDEs). These terminologies reflect the historical development of the two frameworks, with DDPMs rooted in discrete Markov chains and SGMs in score-matching and continuous-time diffusion processes.

In this thesis, both frameworks are unified under the term "diffusion models". This unified terminology highlights the fundamental insight that DDPMs and SGMs represent two perspectives of the same underlying framework, differing only in their treatment of time as discrete or continuous. As will be shown and discussed, these frameworks are mathematically equivalent in the limit of infinite time steps. Adopting this perspective highlights their shared

theoretical foundation, bridging the discrete and continuous-time paradigms, and provides a cohesive framework for exploring their principles and applications throughout this work.

Forward Process. The forward process in diffusion and score-based generative models progressively corrupts the original data into noise, providing the foundation for learning the reverse process that generates data from noise. The design of the forward process ensures that over time, the data distribution transitions smoothly into a simple, analytically tractable distribution π, such as $\mathcal{N}(0, \mathbf{I})$.

- **DDPM (Discrete Time):**

  In the Denoising Diffusion Probabilistic Model (DDPM), the forward process is a Markov chain that adds Gaussian noise to the data at discrete time steps $t = 1, 2, \ldots, N$. The transition kernel is given by:

  $$q_{X_t|X_{t-1}}(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\alpha_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right),$$

  where:

  - $\alpha_t = 1 - \beta_t$, the scaling factor applied to $\mathbf{x}_{t-1}$,
  - $\beta_t \in (0, 1)$, the predefined variance schedule controlling the amount of noise at step t,

  This formulation allows the process to be expressed in terms of the initial data $\mathbf{x}_0$:

  $$q_{X_t|X_0}(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}\right),$$

  where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ is the cumulative product of α_s. This closed-form expression simplifies training and allows direct sampling from any step t.

- **SGM (Continuous Time):** The forward process in Score-Based Generative Models (SGM) is described by a continuous-time Stochastic Differential Equation (SDE) that progressively perturbs the data distribution with infinitesimal noise:

  $$dX_t = f(X_t, t)\,dt + g(t)\,dW_t,$$

  where:

  - W_t is a standard Wiener process (Brownian motion),
  - $f(X_t, t)$ is the drift coefficient, defining the deterministic evolution of the data distribution,

– $g(t)$ is the diffusion coefficient, controlling the scale of the stochastic noise.

The drift and diffusion coefficients $f(X_t,t)$ and $g(t)$ are carefully chosen so that, after a sufficiently large diffusion time T, the data distribution evolves into a distribution that is very close to a simple, analytically tractable distribution, typically the standard normal distribution $\mathcal{N}(\mathbf{0},\mathbf{I})$.

Popular choices for the forward SDE include:

– **Variance Preserving (VP) SDE:**

$$f(X_t,t) = -\frac{1}{2}\beta(t)X_t, \quad g(t) = \sqrt{\beta(t)},$$

where $\beta(t)$ is a predefined noise schedule that controls the rate of diffusion. This choice preserves the variance of the data distribution throughout the process. The limiting distribution as $t \to \infty$ is the standard normal distribution. In practice, after sufficient diffusion time T, the perturbed distribution $p_{X_T} \approx \mathcal{N}(\mathbf{0},\mathbf{I})$.

– **Variance Exploding (VE) SDE:**

$$f(X_t,t) = \mathbf{0}, \quad g(t) = \sqrt{\frac{d[\sigma^2(t)]}{dt}},$$

where $\sigma(t)$ is a monotonically increasing function that scales the noise over time. In this formulation, the variance of the data distribution "explodes." After sufficient diffusion time T, the perturbed distribution $p_{X_T} \approx \mathcal{N}(\mathbf{0},\sigma(T)^2\mathbf{I})$.

These formulations ensure a smooth progression from the data distribution to the prior, providing a well-defined framework for the reverse process to reconstruct data from noise.

Reverse Process. The reverse process reconstructs the original data distribution by inverting the forward process, progressively removing noise to generate data samples:

- **DDPM (Discrete Time):** The reverse process in DDPMs is a Markov chain that progressively denoises the data at each step. The conditional distribution for each step is expressed as:

$$p_{X_{t-1}|X_t,\theta}(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t,t), \beta_t^2\mathbf{I}\right),$$

where $\mu_\theta(\mathbf{x}_t,t)$ is a neural network that predicts the mean of the denoised distribution, given the noisy input $\mathbf{x}_t$ and the timestep t. Sampling is performed iteratively by drawing from $p_{X_{t-1}|X_t,\theta}(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$, starting with $\mathbf{x}_T \sim \pi = \mathcal{N}(\mathbf{0},\mathbf{I})$.

- **SGM (Continuous Time):** In SGMs, the reverse process is governed by the reverse-time SDE, derived from Anderson's theorem [1]:

$$dX_t = \left[f(X_t,t) - g(t)^2 \nabla_{X_t} \log p_{X_t}(X_t) \right] dt + g(t)\, d\bar{W}_t,$$

  where:

  - $\nabla_{X_t} \log p_{X_t}(X_t)$ is the gradient of the log density (score function) of the perturbed distribution at diffusion time t. In practice, the score function is approximated by a neural network $s_\theta(X_t,t)$, referred to as the score model,
  - $\bar{W}_t$ is a reverse-time Wiener process.

  Sampling proceeds by solving this reverse-time SDE, starting from $X_T \sim \pi \approx p(X_T)$ (e.g., $\mathcal{N}(\mathbf{0},\mathbf{I})$) and integrating backward from $t = T$ to $t = 0$. This iterative process generates high-quality samples that closely follow the original data distribution.

The reverse process lies at the core of diffusion-based generative modeling, enabling the transformation of simple noise into complex data distributions while preserving high fidelity.

TRAINING OBJECTIVE. Training involves learning to approximate the reverse process by minimizing specific loss functions:

- **DDPM:** In the DDPM framework, the training objective minimizes an expression that upper bounds the negative log-likelihood of the data. This objective is defined as:

$$\mathscr{L}_{\text{DDPM}}(\theta) = \sum_{t=1}^{T} \mathbb{E}_{X_0 \sim P(X_0),\, \mathbf{E}_t \sim \mathcal{N}(\mathbf{0},\mathbf{I})} \left[\left\| \mathbf{E}_t - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_t}\, X_0 + \sqrt{1-\bar{\alpha}_t}\, \mathbf{E}_t, t \right) \right\|^2 \right],$$

  where:

  - $\mathbf{E}_t \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ represents Gaussian noise added at time step t,
  - $\varepsilon_\theta(\mathbf{x}_t,t)$ is a neural network that estimates the noise component in $\mathbf{x}_t$,
  - $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ is the cumulative product of the noise schedule parameters,
  - $\alpha_t = 1 - \beta_t$, where β_t is the variance of the noise added at each time step.

  By minimizing $\mathscr{L}_{\text{DDPM}}(\theta)$, we effectively minimize an upper bound on the negative log-likelihood, enhancing the model's capacity to generate realistic data samples.

The mean μ_θ of the reverse process used during sampling is derived from the predicted noise ε_θ as follows:

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

where:

- $\hat{\mathbf{x}}_0 = \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}}$ is the reconstructed noiseless data point.

This formulation ensures that the reverse diffusion process effectively removes noise at each step, gradually transforming pure noise into high-quality data samples during generation.

- **SGM:** In the Score-Based Generative Modeling (SGM) framework, the neural network s_θ is trained to approximate the *score function* of the perturbed data distribution $P(X_t)$—that is, the gradient of its log-density $\nabla_{\mathbf{x}_t} \log p_{X_t}(\mathbf{x}_t)$. This is achieved by minimizing the weighted **Denoising Score Matching** objective:

$$\mathscr{L}_{\mathrm{DSM}}(\theta) = \frac{1}{2}\int_0^T \lambda(t)\,\mathbb{E}_{(X_0,X_t)\sim P(X_0,X_t)}\left[\left\| s_\theta(X_t,t) - \nabla_{X_t}\log p_{X_t|X_0}(X_t \mid X_0)\right\|^2\right] dt,$$

where:

  - $p_{X_t|X_0}(\mathbf{x}_t \mid \mathbf{x}_0)$ is the perturbation kernel of the forward Stochastic Differential Equation (SDE),
  - $\lambda(t)$ is a weighting function,
  - $\mathbf{x}_t$ is the diffused data point at time t.

Song et al. [85] showed that for the particular choice $\lambda(t) = g(t)^2$, where $g(t)$ is the diffusion coefficient of the forward SDE, the $\mathscr{L}_{\mathrm{DSM}}$ objective serves as an upper bound on the negative log-likelihood.

In practice, a different weighting function—often referred to as the *simple weighting*—is commonly used:

$$\lambda(t) = \sigma(t)^2,$$

where $\sigma(t)^2$ denotes the variance of the forward perturbation kernel $p_{X_t|X_0}(\mathbf{x}_t \mid \mathbf{x}_0)$. This weighting function has been found to yield superior generative performance, particularly in the image domain where it was initially tested. Consequently, it has become the *de facto* standard for practical implementations.

EQUIVALENCE BETWEEN DDPM AND SGM. DDPMs and SGMs are equivalent in the limit as the number of timesteps $N \to \infty$. Specifically, the forward process in a DDPM converges to the Variance Preserving Stochastic Differential Equation (VP-SDE) used in SGMs:

$$dX_t = -\tfrac{1}{2}\beta(t)X_t\,dt + \sqrt{\beta(t)}\,dW_t,$$

where $\beta(t)$ is the continuous noise schedule and W_t is a standard Wiener process. This result is detailed in Song et al.'s paper *Score-Based Generative Modeling through Stochastic Differential Equations* (2020).

PROBABILITY FLOW ODE. An alternative to the stochastic reverse-time SDE is the **probability flow ODE**, which provides a deterministic mapping between the data distribution and the prior. The probability flow ODE is defined as:

$$\frac{d\mathbf{x}_t}{dt} = f(\mathbf{x}_t,t) - \frac{1}{2}g(t)^2 s_\theta(\mathbf{x}_t,t),$$

where:

- $f(\mathbf{x}_t,t)$ and $g(t)$ are the drift and diffusion coefficients of the forward SDE,
- $s_\theta(\mathbf{x}_t,t)$ is the approximation of the score function at time t.

Under the assumption of a perfectly approximated score function, the probability flow ODE shares the same marginal probability distributions as the reverse-time SDE for all diffusion times. This equivalence arises from considering the Fokker-Planck equations corresponding to both processes, which govern the time evolution of probability densities.

Computational Efficiency. In practice, integrating the probability flow ODE has been observed to be computationally more efficient than simulating the reverse-time SDE. Fewer integration steps are typically required to achieve similar sample quality, making it a preferred choice for sampling in diffusion models. This efficiency stems from the deterministic nature of the ODE, which allows for larger step sizes, unlike the stochastic SDE that often requires smaller step sizes to maintain sample fidelity.

Likelihood Computation with the Probability Flow ODE. The probability flow ODE enables likelihood computation via the instantaneous change of variables formula. Given the score function $s_\theta(\mathbf{x}_t,t)$, the log-likelihood of a data point $\mathbf{x}_0$ can be expressed as:

$$\log p_{X_0}(\mathbf{x}_0) = \log p_{X_T}(\mathbf{x}_T) + \int_0^T \nabla_{\mathbf{x}_t} \cdot \mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t,t)\,dt,$$

where $\mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t, t)$ is the drift term of the probability flow ODE. Direct computation of the divergence term $\nabla_{\mathbf{x}_t} \cdot \mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t, t)$ is often computationally expensive. To address this, the Skilling-Hutchinson trace estimator is used:

$$\nabla_{\mathbf{x}_t} \cdot \mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t, t) \approx \mathbb{E}_{\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left[\varepsilon^\top \nabla_{\mathbf{x}_t} \mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t, t) \varepsilon \right],$$

where $\nabla_{\mathbf{x}_t} \mathbf{f}_\theta^{\mathrm{ODE}}(\mathbf{x}_t, t)$ is the Jacobian of $\mathbf{f}_\theta^{\mathrm{ODE}}$. The term inside the expectation can be efficiently computed using Jacobian-vector product calculations, making this approach both practical and scalable for accurate likelihood computation.

CONDITIONAL GENERATION. Diffusion models can be naturally extended to conditional generation tasks, where the objective is to model the conditional distribution $p(X_0 \mid C)$, with C representing the conditioning variable (e.g., class labels, textual descriptions, or other modalities). This extension enables generating data conditioned on specific information, with applications such as super-resolution, inpainting, and text-guided generation.

A widely used approach for conditional generation in diffusion models is the weighted **Conditional Denoising Score-Matching** (CDSM) objective. The core idea is to train a neural network $s_\theta(\mathbf{x}_t, \mathbf{c}, t)$ that takes as input the noisy data $\mathbf{x}_t$, the condition $\mathbf{c}$, and the time t, and outputs an estimate of the conditional score $\nabla_{\mathbf{x}_t} \log p_{X_t|C}(\mathbf{x}_t \mid \mathbf{c})$. The training objective is defined as:

$$\mathscr{L}_{\mathrm{CDSM}}(\theta) = \frac{1}{2} \int_0^T \mathbb{E}_{(X_0, X_t, C) \sim p(X_0, X_t, C)} \left[\lambda(t) \left\| s_\theta(X_t, C, t) - \nabla_{X_t} \log p_{X_t|X_0}(X_t \mid X_0) \right\|^2 \right] dt. \tag{1.1}$$

The only difference between the DSM objective used for unconditional generation and the CDSM objective used for conditional generation lies in the incorporation of the conditioning variable C into the score model. Note that the regression target—the score function of the perturbation kernel—remains the same. While early empirical results were promising, it was unclear why the minimizer of the CDSM objective approximates the true conditional score function. In Chapter **??**, we provide the first formal proof of consistency for the CDSM estimator, establishing its theoretical validity and confirming its utility in modeling conditional distributions.

STRENGTHS. Diffusion-based generative models have emerged as a powerful class of approaches for learning complex data distributions because they exhibit several notable strengths highlighted across the literature. First, they circumvent many of the pitfalls associated with adversarial training by relying on well-defined, non-adversarial objectives. This design choice reduces issues like mode collapse and training instability that often arise

with GANs, leading to more robust and reliable convergence behavior. Second, diffusion models have demonstrated excellent sampling performance in diverse domains—including image, audio, video, and molecule generation—primarily because they do not hinge on stringent architectural constraints (as seen with normalizing flows) or adversarial losses (as with GANs). Their ability to flexibly capture multi-modal and high-dimensional distributions makes them especially appealing when dealing with complex, real-world datasets. Moreover, they derive from well-established stochastic process theory, offering a solid mathematical foundation that guides both theoretical analysis and practical design choices.

WEAKNESSES. However, one of the most frequently cited weaknesses of diffusion models is their relatively slow sampling speed. The underlying process typically entails running a series of iterative refinement steps—often in the hundreds or thousands—to gradually remove noise and produce high-fidelity samples. This iterative nature translates into significant computational overhead, making diffusion models less suitable for time-critical applications where rapid or real-time generation is paramount. Furthermore, while various techniques have been proposed to reduce sampling time (such as employing fewer refinement steps or adopting improved solvers), these methods usually introduce trade-offs between sample quality and speed.

ADVANCEMENTS IN SAMPLING SPEED. Significant progress has been made in accelerating the sampling speed of diffusion models, effectively addressing their primary limitation of slow generation due to iterative steps. A critical observation in this area is that the probability flow ODE, which shares the same marginal distribution as the reverse SDE, is generally preferred for faster sampling. The ODE formulation tends to produce better samples than the corresponding SDE in the low discretization regime [86]. Consequently, many methods that accelerate diffusion models focus on learning to efficiently integrate the probability flow ODE, often as a post-training optimization.

Techniques such as **Denoising Diffusion Implicit Models (DDIM)** [83] reformulate the reverse diffusion process by utilizing non-Markovian dynamics, significantly reducing the number of required sampling steps without compromising sample quality. **Latent Diffusion Models (LDMs)** [74] enhance efficiency by performing diffusion in a compressed latent space, drastically decreasing computational demands while preserving high-resolution outputs. **Progressive Distillation** [79] further compresses the multi-step diffusion process into models capable of generating high-fidelity samples in significantly fewer steps—sometimes even a single step. Recently, **Consistency Models** [84] have been introduced, directly mapping noise to data in one step while retaining the advantages of iterative methods.

Collectively, these advancements have significantly reduced sampling times and made diffusion models practical for real-time and interactive applications across various domains.

1.2 Self-Supervised Learning

1.2.1 Motivation

In today's data-rich environment—spanning social media, sensor outputs, and extensive collections of images, videos and text—the primary challenge is not the lack of data but our ability to utilize it effectively. Despite the vast amounts of information available, much remains unlabeled and underexploited due to the high costs and impracticalities associated with manual labeling. This situation poses a crucial question: *how can we unlock the potential of unlabeled data to develop intelligent systems that learn, adapt, and generalize in a manner akin to human learning?*

Self-supervised learning (SSL) emerges as a compelling approach to this challenge. Referred to by Yann LeCun as the "dark matter of intelligence" [49], SSL enables machines to learn from the abundant unlabeled data by exploiting the inherent structures within it. LeCun remarks:

> "Common sense helps people learn new skills without requiring massive amounts of teaching for every single task. Self-supervised learning is one of the most promising ways to build such background knowledge and approximate a form of common sense in AI systems."

Analogous to how a child learns about the world through observation and pattern recognition rather than explicit instruction, SSL allows models to derive meaningful representations from data without manual labels. The transformative impact of SSL is evident across multiple domains:

Natural Language Processing (NLP): SSL has transformed NLP by enabling models to understand and generate human language with improved contextual awareness. Early models like Word2Vec [56] and GloVe [65] learned semantic relationships between words through context prediction. Advanced models such as BERT [22] and GPT [68] employed masked language modeling to pretrain on extensive text corpora, enhancing capabilities in tasks like text completion and machine translation.

Computer Vision (CV): In CV, SSL techniques like Contrastive Predictive Coding (CPC) [62], SimCLR [18], and MoCo [36] have enabled models to learn visual representations without reliance on labeled datasets. For example, Facebook's SEER model [32], trained on a billion unlabeled images, achieved state-of-the-art performance on ImageNet, illustrating the efficacy of learning from unlabeled visual data.

Speech Recognition: SSL models such as wav2vec [80] and wav2vec 2.0 [5] have significantly advanced speech recognition by learning representations from raw audio data.

By predicting future audio samples from past ones, these models capture essential phonetic and linguistic features, improving transcription accuracy and the handling of diverse accents and intonations.

Multimodal Learning: SSL has significantly advanced the integration of multiple data modalities, enabling AI systems to learn joint representations that bridge the gap between domains like vision, language, and audio. Pioneering models such as CLIP [67] have shown remarkable success in aligning images and text within a shared latent space, enabling a wide range of applications. These include image captioning, where the model generates descriptive text for images; text-based image retrieval, allowing users to find images using natural language queries; and zero-shot classification, where the model classifies images into categories it has not seen during training, guided only by textual descriptions.

These advancements are not merely incremental; they signify a fundamental shift in machine learning methodologies. By leveraging unlabeled data, SSL enables models to develop a form of common sense, understanding context and patterns previously inaccessible through supervised learning alone.

The potential applications of SSL are extensive. It opens pathways to developing AI systems capable of understanding and generating human-like language and vision, and learning new tasks with minimal supervision. Self-supervised learning represents a significant step toward creating AI that learns and adapts more like humans, enhancing intelligence and adaptability in artificial systems.

1.2.2 Mathematical Formulation of SSL

Let $\mathscr{D} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^d$ be a dataset of unlabeled samples, assumed to be drawn i.i.d. from an unknown distribution $P(X)$ over $\mathbb{R}^d$. The goal of self-supervised learning is to learn an *encoder* function

$$f_\theta : \mathbb{R}^d \to \mathbb{R}^k,$$

where $k \leq d$, that maps each input $\mathbf{x} \in \mathbb{R}^d$ into a latent representation $\mathbf{h} = f_\theta(\mathbf{x}) \in \mathbb{R}^k$. These lower-dimensional embeddings act as a concise, informative summary of the original data, improving generalization and serving as a foundation for downstream tasks (e.g., classification, clustering, retrieval).

The encoder f_θ is parameterized by θ and is trained via a *self-supervised loss* function

$$\theta^* \;=\; \arg\min_\theta \mathbb{E}_{X\sim P(X)}\big[\mathscr{L}(X;\theta)\big].$$

The exact form of $\mathscr{L}$ depends on the specific SSL approach and typically encourages invariance to augmentations or alignment among related samples. In practice, it may include two or more terms (e.g., a contrastive component plus a regularization term) to jointly enforce closeness of similar samples and separation of dissimilar ones in the latent space.

1.2.3 Generative vs. Discriminative SSL

Self-supervised learning methods can be broadly categorized into two paradigms: **generative** and **discriminative** approaches. Understanding the strengths and applications of each is crucial for selecting appropriate methods for specific tasks.

Generative Approaches

These methods aim to extract meaningful latent representations by modeling the underlying data distribution. These approaches can be broadly categorized into two types based on the scope of the distribution being modeled:

1. Full data distribution $P(X)$
2. Masked data distribution $P(X_{\text{masked}}|X_{\text{context}})$

Learning Latent Representations by Modeling the Full Distribution $p(X)$

- *Auto-Regressive (AR) Models*: AR models factorize the data distribution into a product of conditional probabilities. These models generate each data element sequentially, conditioned on the previous ones.
  - *GPT and GPT-2* [68]: Learn language representations by predicting the next token given preceding context, enabling robust text generation and understanding. GPT embeddings serve as inputs for fine-tuned classifiers in NLP tasks.
  - *PixelRNN and PixelCNN* [61]: Model pixel dependencies in images for self-supervised representation learning, useful for tasks like image completion. PixelCNN representations have been used in image segmentation.
- *Normalizing Flows*: These models transform a simple latent distribution into a complex target distribution using a sequence of invertible mappings, preserving exact likelihoods. Examples include:

  - *FlowGMM* [42]: Combines normalizing flows with a Gaussian mixture model in the latent space to extract representations useful for tasks such as text classification, tabular data analysis, and semi-supervised image classification.

- *Auto-Encoding (AE) Models*: Autoencoders reconstruct data from latent representations, learning robust embeddings for various downstream tasks. Examples include:

  - *Variational Autoencoders (VAEs)* [46]: Introduce probabilistic latent spaces, learning smooth representations useful for clustering and semi-supervised learning.
  - *VQ-VAE* [63]: Extends VAEs to discrete latent spaces, providing meaningful representations that have been widely applied in downstream tasks:
    * In *audio processing*, VQ-VAE representations have been used in *wav2vec* [5] for unsupervised pretraining on speech data, enabling improvements in downstream speech recognition tasks by learning phonetic and linguistic features from raw audio.
    * In *vision*, the latent codes are inputs to autoregressive models for image generation, inpainting, and enhancement tasks [70].
    * In *multimodal learning*, VQ-VAE representations bridge modalities such as text and images, aiding in tasks like image captioning and cross-modal retrieval [24].

- *GAN-Based Models*: Generative Adversarial Networks (GANs) learn latent representations by training a generator and a discriminator adversarially. Recent advancements adapt GANs for representation learning. Examples include:

  - *StyleGAN* [44]: Generates high-quality images with interpretable latent spaces that enable fine-grained editing (e.g., facial expressions and lighting adjustments). StyleGAN embeddings are widely used in image synthesis and manipulation tasks.
  - *VQ-GAN* [26]: Combines vector quantization and adversarial training to learn discrete latent representations. These representations are used for tasks such as text-to-image generation, image compression, and cross-modal retrieval.

Learning Latent Representations by Modeling the Masked Distribution $p(X_{\textbf{masked}}|X_{\textbf{context}})$

- **Masked Image Modeling:** Models learn representations by reconstructing occluded parts of an image, leveraging context to infer the missing information.

  - *Examples:*
    - **BEiT** [6]: Learns representations for downstream tasks like image classification and segmentation by reconstructing masked patches of images.
    - **Masked Autoencoders (MAE)** [35]: Focuses on learning efficient representations by masking a large portion of the input image and reconstructing it from the unmasked portion.
  - *Downstream Applications:* Representations from masked image models are particularly useful for visual tasks such as object detection, segmentation, and classification, where capturing spatial relationships is critical.

- **Masked Language Modeling:** In NLP, masked language models predict missing words or tokens in a sequence, learning rich contextual embeddings.
  - *Examples:*
    - **BERT** [22]: A foundational model in NLP, pretraining on masked tokens enables fine-tuning for tasks such as sentiment analysis and named entity recognition.
    - **RoBERTa** [54]: Builds on BERT by optimizing training dynamics and is widely used for text classification and question answering.
  - *Downstream Applications:* Masked language model representations are extensively used for text-based tasks, including translation, summarization, and retrieval.

Discriminative Approaches

These methods focus on learning robust feature representations by solving pretext tasks or leveraging contrastive objectives without explicitly modeling the data distribution. The main motivation is to learn representations that are useful for downstream tasks by capturing essential features and invariances in the data. Key techniques include:

- **Contrastive Learning**:
  - *Motivation and Logic*: Contrastive learning aims to learn robust feature representations by maximizing agreement between similar pairs (positive pairs) and minimizing agreement between dissimilar pairs (negative pairs). The central idea is to structure the latent space such that representations of similar samples are closer, while representations of different samples are farther apart.

– *Generalized Contrastive Loss*: [95] unified contrastive losses under a general framework, where the loss is expressed as:

$$\mathscr{L}_{\phi,\psi}(\theta) = \sum_{i=1}^{N} \phi\left(\sum_{j\neq i} \psi\left(\|\mathbf{z}_i - \mathbf{z}_j\|_2^2 - \|\mathbf{z}_i - \mathbf{z}_i'\|_2^2\right)\right),$$

where:

* $\mathbf{z}_i$ and $\mathbf{z}_i'$ are the representations of a sample i and its positive pair (augmented view of the same sample).
* $\|\mathbf{z}_i - \mathbf{z}_j\|_2^2$ is the squared distance between the representations of sample i and a negative sample j.
* ϕ and ψ are monotonically increasing and differentiable scalar functions that control how positive and negative distances are weighted.

Specific choices of ϕ and ψ lead to well-known contrastive learning methods:

* **InfoNCE Loss** [62]: Setting $\phi(x) = \tau\log(\varepsilon + x)$ and $\psi(x) = \exp(x/\tau)$ results in:

$$\mathscr{L}_{\text{InfoNCE}} = -\tau\sum_{i=1}^{N}\log\frac{\exp\left(-\|\mathbf{z}_i - \mathbf{z}_i'\|_2^2/\tau\right)}{\varepsilon\exp\left(-\|\mathbf{z}_i - \mathbf{z}_i'\|_2^2/\tau\right) + \sum_{j\neq i}\exp\left(-\|\mathbf{z}_i - \mathbf{z}_j\|_2^2/\tau\right)}.$$

* **SimCLR Loss** [18]: Simplifies InfoNCE by setting $\varepsilon = 0$, removing the stabilization term in the denominator.

- **Self-Distillation**:

– *Motivation and Logic*: These methods aim to learn representations without the need for negative samples, addressing issues like the need for large batch sizes or memory banks in contrastive learning. The model learns by predicting its own representations under different augmentations, promoting consistency.

– *Training Objective*: The model minimizes a prediction loss between representations of different augmented views. For instance, in BYOL, the loss is:

$$\mathscr{L}_{\text{BYOL}} = \left\|\mathbf{q}_i - \text{stopgrad}(\mathbf{z}_i^+)\right\|_2^2,$$

where:

* $\mathbf{q}_i = f_\theta(\mathbf{x}_i)$ is the output of the online network.
* $\mathbf{z}_i^+ = f_{\theta'}(\mathbf{x}_i^+)$ is the output of the target network.

    * stopgrad indicates that gradients are not backpropagated through this path.
    * θ' is an exponential moving average of the parameters θ.
  - *Examples*:
    * *BYOL* [34]: Uses an online and a target network to predict one view from another, with the target network updated via an exponential moving average.
    * *SimSiam* [19]: Simplifies BYOL by removing the momentum encoder and employing a stop-gradient operation to prevent representational collapse.

- **Feature Decorrelation Methods**:
  - *Motivation and Logic*: Feature decorrelation methods aim to learn rich and diverse representations by reducing redundancy among feature dimensions. Inspired by Canonical Correlation Analysis (CCA), these approaches encourage each component of the embedding to capture unique information, promoting uncorrelated and informative features. By focusing on decorrelating feature dimensions, these methods prevent representational collapse and enhance the quality of learned representations without relying on negative samples or complex training schemes.
  - *Training Objective*: The central idea is to make embeddings from different augmented views of the same data similar (invariance) while ensuring that different feature dimensions are uncorrelated (decorrelation). This is achieved by combining an invariance term with a decorrelation term in the loss function, encouraging both similarity across views and diversity among features.
    * *Barlow Twins* [101] employs two identical networks to process different augmented views of the same image. It computes a cross-correlation matrix $\mathbf{C}$ between the embeddings from the two networks. The loss function encourages the diagonal elements of $\mathbf{C}$ to be close to one (promoting invariance) and the off-diagonal elements to be close to zero (reducing redundancy):

$$\mathscr{L}_{\mathrm{BT}} = \sum_{i=1}^{d} (1 - \mathbf{C}_{ii})^2 + \lambda \sum_{i \neq j} \mathbf{C}_{ij}^2.$$

      This approach enables the learning of features that are invariant to augmentations while ensuring each dimension captures distinct information.
    * *VICReg* [7] introduces a loss function with three components. The invariance term minimizes the mean squared error between embeddings from different views. The variance term ensures the standard deviation of each feature dimension exceeds a threshold to prevent collapse. The covariance term

penalizes off-diagonal covariance elements to reduce redundancy. The loss is expressed as:

$$\mathscr{L}_{\text{VICReg}} = \underbrace{\frac{1}{B}\sum_{b=1}^{B} \|\mathbf{z}_b^{(1)} - \mathbf{z}_b^{(2)}\|_2^2}_{\text{Invariance}} + \underbrace{\gamma \sum_{j=1}^{d} \max(0, s - \sigma(\mathbf{z})_j)}_{\text{Variance}} + \underbrace{\mu \sum_{i \neq j} \text{Cov}(\mathbf{z})_{ij}^2}_{\text{Covariance}},$$

where γ and μ balance the terms, s is the variance threshold, and $\text{Cov}(\mathbf{z})_{ij}$ is the off-diagonal element of the covariance matrix of $\mathbf{z}$.

- **Clustering-Based Methods**:
  - *Motivation and Logic*: These approaches leverage unsupervised clustering to group similar data points, learning representations that capture group-level semantic structures. The model alternates between clustering the data in the representation space and updating the encoder to produce features that align with these clusters.
  - *Training Objective*: The encoder is trained to predict pseudo-labels derived from cluster assignments. The loss function typically involves cross-entropy between the predicted labels and the pseudo-labels:

$$\mathscr{L}_{\text{cluster}} = \sum_{i=1}^{N} \ell_{\text{CE}}(f_\theta(\mathbf{x}_i), y_i),$$

    where y_i is the cluster assignment for $\mathbf{x}_i$ obtained via a clustering algorithm (e.g., k-means), and ℓ_{CE} denotes the cross-entropy loss.
  - *Examples*:
    - *DeepCluster* [16]: Performs k-means clustering on the learned features and uses the cluster assignments as pseudo-labels for training.
    - *SwAV* [17]: SwAV uses siamese networks to generate embeddings for two augmented views of an image and aligns them with trainable prototypes. It computes cluster assignments online using the Sinkhorn-Knopp algorithm and minimizes cross-entropy between these assignments and predicted probabilities, enabling efficient training without pairwise comparisons.
- **Pretext Task-Based Methods**

- *Motivation and Logic*: Pretext task-based methods aim to learn useful representations by training models on auxiliary tasks where the labels can be derived automatically from the data itself. These tasks, often unrelated to the downstream tasks, encourage the model to capture meaningful patterns and structures in the data that generalize well across applications. By solving these simple yet informative tasks, the model learns features that are transferable to various downstream tasks without requiring manual annotations.
- *Examples*:
  - **Rotation Prediction** [29]: The model learns representations by predicting the rotation angle applied to an image. Given an image rotated by one of four angles (0°, 90°, 180°, or 270°), the model classifies the rotation angle. This task encourages the model to understand spatial relationships and object structures within the image.
  - **Jigsaw Puzzle Solving** [59]: The model is tasked with solving jigsaw puzzles by predicting the correct arrangement of shuffled image patches. This pretext task forces the model to learn contextual and semantic relationships between different parts of an image.
  - **Temporal Ordering** [57]: For video data, the model learns representations by predicting the correct temporal order of video frames. This task helps the model capture temporal dependencies and motion dynamics in videos.

Choosing Between Generative and Discriminative SSL

The choice between generative and discriminative self-supervised learning methods depends on the specific objectives and requirements of the task. Each approach has its strengths and is more suitable under certain circumstances.

Generative SSL is more useful when: Generative SSL methods excel in scenarios that demand a fine-grained understanding of the data distribution. These methods capture detailed and nuanced information within the extracted representations, making them ideal for tasks that require precise and intricate information about the data. For instance, in image editing and manipulation, generative models such as StyleGAN [44, 45] enable sophisticated modifications by navigating the latent space. By solving optimization problems to move in specific directions within this space, attributes like facial expressions, styles, or lighting conditions can be altered, providing a level of fine-grained control that discriminative models typically cannot offer. Additionally, generative SSL is advantageous in applications like anomaly

detection, where understanding the complete data distribution allows for the identification of outliers, and in cross-modal tasks such as text-to-image generation, where translating between different data modalities requires comprehensive distribution modeling.

Discriminative SSL is more useful when: Discriminative SSL methods are preferable when the primary goal is to learn efficient, task-relevant representations tailored for specific downstream applications. These methods optimize embeddings to effectively distinguish between different classes or instances, making them highly suitable for tasks such as classification and retrieval. For example, in image classification, discriminative models like SimCLR [18] and MoCo [36] produce embeddings that enhance performance by focusing on feature discrimination. Furthermore, discriminative SSL is advantageous in environments with limited computational resources, as these methods often require fewer resources and are more scalable. They are also ideal for real-time or latency-sensitive applications, where fast inference and low latency are critical, such as online recommendation systems or real-time monitoring.

Blurring the Lines: Hybrid Approaches

The distinction between generative and discriminative SSL methods is increasingly becoming blurred, with hybrid approaches emerging that combine the strengths of both paradigms. These hybrid models leverage the fine-grained distribution modeling of generative methods alongside the efficient, task-specific representation learning of discriminative methods. Notable examples include BEiT [6] and iBOT [103], which integrate masked image modeling (a generative task) with self-distillation objectives (a discriminative task). By doing so, these models capture rich data representations while maintaining computational efficiency, effectively addressing a broader range of tasks.

1.2.4 Utilizing Learned Representations in Downstream Tasks

Once the representation function f_θ is learned, the representations $\mathbf{h}_i = f_\theta(\mathbf{x}_i)$ can be utilized in various downstream tasks. These representations capture essential features and structures in the data, making them valuable inputs for different applications.

Classification

In classification problems, the goal is to assign input data to one of several predefined categories. The learned representations $\mathbf{h}_i$ serve as informative features that can be used to train a classifier. Specifically, we can define a classifier $g_\phi : \mathbb{R}^k \to \mathbb{R}^C$, where C is the number

of classes, and ϕ represents the parameters of the classifier (e.g., weights of a fully connected layer).

The classifier is trained on a labeled dataset $\{(\mathbf{x}_j, y_j)\}_{j=1}^{M}$, where $y_j \in \{1, 2, \ldots, C\}$ is the class label for sample $\mathbf{x}_j$. The training involves minimizing a supervised loss function, typically the cross-entropy loss:

$$\phi^* = \arg\min_{\phi} \frac{1}{M} \sum_{j=1}^{M} \ell_{\text{CE}}(g_\phi(\mathbf{h}_j), y_j),$$

where $\mathbf{h}_j = f_\theta(\mathbf{x}_j)$ and ℓ_{CE} denotes the cross-entropy loss function.

The representations $\mathbf{h}_i$ often capture high-level semantic features and are invariant to variations in the input data that are irrelevant to the classification task, such as changes in lighting, orientation, or background. This invariance leads to better generalization and robustness, improving classification performance even when labeled data is limited.

In practice, the representations can be utilized in two ways:

- **Fixed Feature Extraction**: The encoder f_θ is kept frozen, and only the classifier g_ϕ is trained on the labeled data. This approach is useful when computational resources are limited or when the labeled dataset is small.

- **Fine-Tuning**: Both the encoder f_θ and the classifier g_ϕ are trained (or fine-tuned) jointly on the labeled data. This allows the model to adapt the learned representations to the specific nuances of the downstream task, often leading to better performance.

Other Downstream Tasks

Similar logic can be applied to other downstream tasks such as object detection, semantic segmentation, instance segmentation, image captioning, and more. In these tasks, the extracted representations $\mathbf{h}_i$ serve as inputs to task-specific prediction heads. For example, in object detection, the representations can be used to extract feature maps that help in localizing and classifying objects within images. In semantic segmentation, the representations provide spatial and contextual information necessary for assigning class labels to each pixel in an image. For image captioning, the representations capture visual features that can be input to sequence models to generate descriptive captions for images.

In all these tasks, the pretrained encoder f_θ acts as a feature extractor, and the task-specific prediction heads are trained to perform the desired task using the representations as input. Similar to classification, the encoder f_θ can be kept fixed or fine-tuned along with the prediction heads. This flexibility allows the model to either leverage the general representations learned during self-supervised pretraining or adapt them to the specific

nuances of the downstream task. This approach utilizes the strengths of self-supervised learning to enhance performance across various domains and applications.

1.2.5 Challenges and Limitations in Current Methodologies

Despite significant advancements, current self-supervised learning (SSL) methods face several challenges that hinder their ability to fully capture the underlying structure and complexity of data. In this section, we discuss key weaknesses—particularly those we aim to address in this work—including intrinsic dimension estimation, limited model expressiveness in variational autoencoders (VAEs), and the lack of geometric understanding and interpretability in the latent space.

Intrinsic Dimension Estimation

While SSL methods strive to extract meaningful representations from data, they often do not make an informed choice about the latent dimensionality of these representations. Standard practice involves selecting a latent dimension based on convention or precedent in the literature—commonly using values like 512—regardless of the dataset's inherent complexity. This arbitrary selection can lead to suboptimal performance.

If the optimal latent dimension is k and a significantly larger dimension is chosen, the model may incorporate redundant information in downstream tasks. This redundancy might adversely affect generalization. Conversely, selecting a latent dimension smaller than k risks losing essential information, leading to reduced performance due to insufficient representation capacity.

Developing methods that can estimate or inform the choice of the intrinsic dimensionality of the data manifold is crucial. Such methods enable models to learn representations that are both efficient and effective, capturing the essential structure of the data without unnecessary complexity.

In Chapter **??**, we present a novel method for estimating the latent dimension using diffusion models trained on the data distribution. We theoretically and experimentally demonstrate that diffusion (or score-based) models inherently encode information about the intrinsic dimensionality. Specifically, we prove that the Jacobian of the score function—modeled by a diffusion or score-based model—captures the low-dimensional intrinsic structure of the data. When data concentrates around a lower-dimensional manifold of dimension k embedded in a higher-dimensional ambient space of dimension d, the singular value decomposition (SVD) of the Jacobian of the score function yields k vanishing singular values. This finding provides a principled approach to estimating the intrinsic dimension k of the data manifold.

Limited Model Expressiveness in Variational Autoencoders

Variational autoencoders are widely used in unsupervised representation learning due to their ability to model complex data distributions. However, standard VAEs suffer from limited model expressiveness, primarily attributed to two key assumptions: the encoding distribution (often assumed to be a unimodal Gaussian) and the decoding distribution (also assumed to be a unimodal Gaussian).

The assumption of a Gaussian decoding distribution can lead to blurry reconstructions, especially when modeling complex distributions such as natural images. This limitation arises because a unimodal Gaussian cannot capture the multimodal and intricate structures present in such data. As a result, the learned representations may lack the fine-grained details necessary for high-quality reconstructions and downstream tasks that require precise information.

In Chapter **??**, we introduce a novel adaptation of the VAE framework called *ScoreVAE*, which addresses this issue by combining a diffusion-time-dependent encoder with an unconditional diffusion model. By employing Bayes' rule for score functions, we analytically derive a robust and flexible model for the reconstruction distribution. Our approach bypasses the unrealistic Gaussian assumption, resulting in a more expressive VAE capable of capturing the complex structures inherent in high-dimensional data like images.

Geometric Understanding and Interpretability of the Latent Space

Another significant limitation of current methodologies is the lack of geometric understanding and interpretability in the latent space. Existing methods often fail to effectively exploit the geometry of the data manifold, leading to representations that may not reflect the true relationships and structures within the data. This lack of interpretability hinders our ability to understand and manipulate the latent representations meaningfully.

In Chapter **??**, we introduce a pullback Riemannian metric induced by the score function, capturing the intrinsic dimensionality and geometry of the data manifold. This enables closed-form geodesics, interpretable autoencoding, and a deeper understanding of the manifold's structure. By adapting the normalizing flow framework with isometry regularization and base distribution anisotropy, our method preserves local manifold properties, facilitates dimensionality reduction, and enhances the interpretability of latent space representations.

Chapter 2

Thesis Outline and Contributions

2.1 CAFLOW: Conditional Autoregressive Flows

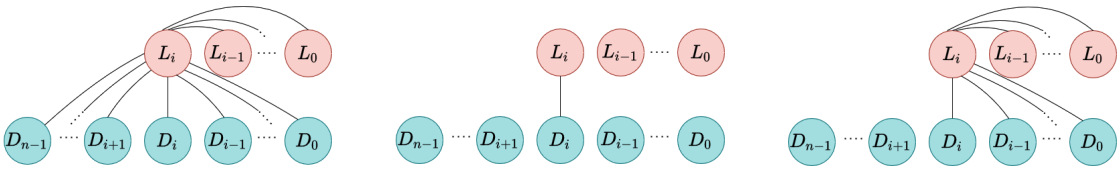


Fig. 2.1 From left to right: ideal dependencies in the i^{th} autoregressive component. Dual-Glow modeling assumption [92]; information is exchanged only between latent spaces having the same dimension. Our modeling assumption; we retain the dependencies between L_i and the latent spaces of lower dimension.

In Chapter **??**, we introduce CAFLOW, a new diverse image-to-image translation model that simultaneously leverages the power of auto-regressive modeling and the modeling efficiency of conditional normalizing flows. We transform the conditioning image into a sequence of latent encodings using a multi-scale normalizing flow and repeat the process for the conditioned image. We model the conditional distribution of the latent encodings by modeling the auto-regressive distributions with an efficient multi-scale normalizing flow, where each conditioning factor affects image synthesis at its respective resolution scale. Our proposed framework performs well on a range of image-to-image translation tasks. It outperforms former designs of conditional flows because of its expressive auto-regressive structure.

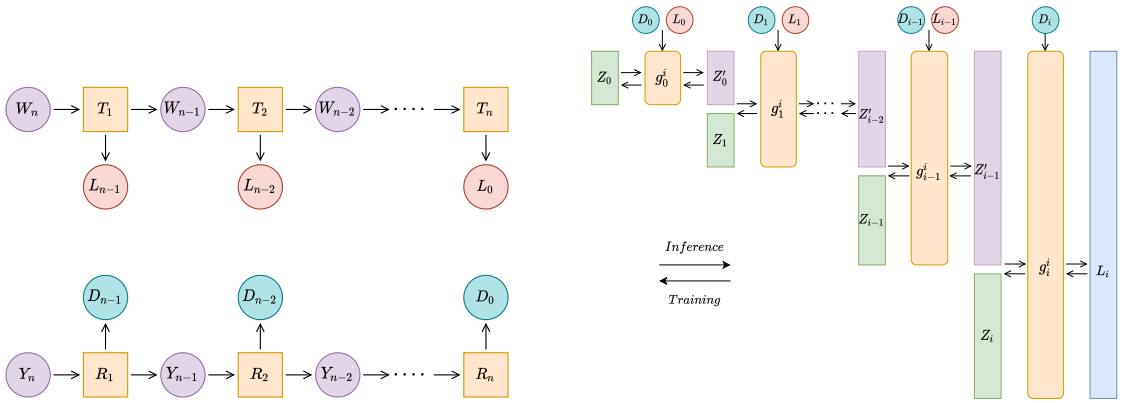


Fig. 2.2 Left: unconditional normalizing flow architecture used to encode conditioning and conditioned images, denoted by $Y_n = Y$ and $W_n = W$ respectively, into a sequence of hierarchical latent variables. Right: design of the conditional transformation G_i^θ that models the i^{th} autoregressive component. The index of the flow i is omitted in both the transformed latent variable Z_j and the intermediate latent variables Z'_j for simplicity.

Originality and Author's Contributions

This chapter is adapted from our published work [8] in the Foundations of Data Science (FoDS) journal. The authors' contributions to this work are as follows:

- **Formulation of Ideas**: The formulation of the CAFLOW framework was entirely my contribution.
- **Experimental Design**: I independently designed all experiments to validate the performance of the proposed framework.
- **Code and Experimental Implementation**: I implemented the entire codebase for the model and conducted all experiments. Christian Etmann provided critical feedback and domain expertise on the training and inference of Normalizing Flows, significantly enhancing the quality of the work.
- **Theory**: I developed the theoretical foundations underpinning the proposed framework, with valuable contributions from Marcelo Carioni, who helped refine and extend the theoretical arguments.
- **Presentation**: Myself and Marcelo Carioni contributed equally to the write-up of this work.

This project was conducted under the supervision of Carola-Bibiane Schönlieb, Christian Etmann, Soroosh Afyouni, and Zoe Kourtzi.

2.2 Non-Uniform Diffusion Models

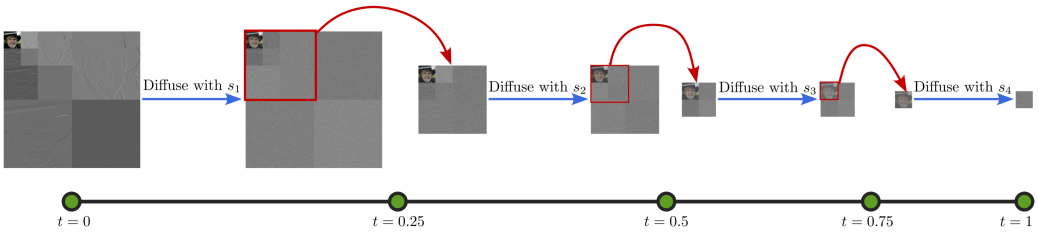


Fig. 2.3 Illustration of a multi-scale diffusion model with three scales. Inspired by multi-scale normalizing flows, this approach diffuses different parts of the image tensor (transformed into multi-level Haar coefficients) at varying speeds. High-frequency detail coefficients diffuse progressively faster, with d_1 diffusing faster than d_2, d_2 faster than d_3, and so on, ensuring that all coefficients reach the same (very low) signal-to-noise ratio (SNR) at their respective terminal diffusion times: $t = 0.25, 0.5, 0.75$, and 1.0, respectively. The low-frequency approximation coefficients a_3 diffuse the slowest, completing their diffusion at $t = 1.0$. The multi-scale structure reduces the dimensionality of the diffusing tensor at each scale, enabling faster computation. Separate neural networks S_1, S_2, S_3, S_4 approximate the score functions at different intervals, leveraging the reduced dimensionality of the intermediate distributions. This hierarchical design mirrors the structure of multi-scale normalizing flows, improving training and sampling efficiency while maintaining high image generation quality.

In Chapter **??**, we introduce *non-uniform diffusion models*. Unlike standard diffusion approaches that apply the same noise injection schedule to every pixel, non-uniform diffusion models allow different pixels (or groups of pixels) to evolve at varying speeds. This flexibility mirrors the hierarchical approach of multi-scale normalizing flows, where transformations occur at multiple scales, enabling the model to capture image structure more efficiently and produce higher-quality samples in less time. By carefully choosing which parts of the image diffuse faster, non-uniform diffusion opens the door to significantly improved performance, both in terms of image fidelity and computational speed.

We demonstrate that non-uniform diffusion models outperform standard uniform diffusion models by achieving superior FID scores in less training time. Furthermore, these models exhibit remarkable efficiency, sampling up to 4.4 times faster at a resolution of 128×128, with even greater speed-ups anticipated at higher resolutions. Leveraging the adaptability of non-uniform diffusion, we introduce the Conditional Multi-Speed Diffusive Estimator (CMDE), a novel approach derived from a specific choice of non-uniform diffusion. CMDE unifies existing methods for conditional score estimation and delivers performance on par with the widely adopted conditional denoising estimator.

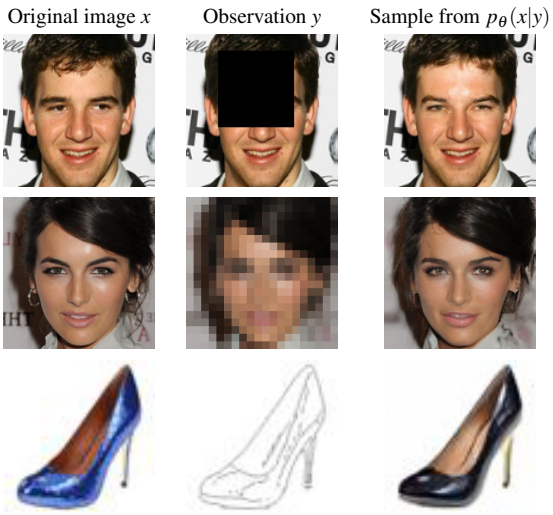


Fig. 2.4 Results from our conditional multi-speed diffusive estimator.

On the theoretical side, we introduce a principled objective for training non-uniform diffusion models and provide a proof of consistency for the conditional denoising estimator, thereby establishing the reliability of the most widely adopted approach to training conditional diffusion models.

Beyond these theoretical and methodological advances, we conduct a comprehensive empirical evaluation of different approaches to training conditional diffusion methods, comparing their effectiveness across tasks such as super-resolution, inpainting, and edge-to-image translation. Finally, to encourage future research and practical adoption, we release MSDiff, an open-source library dedicated to non-uniform and conditional diffusion models, enabling other researchers and practitioners to easily experiment with and build upon our work.

Originality and Author's Contributions

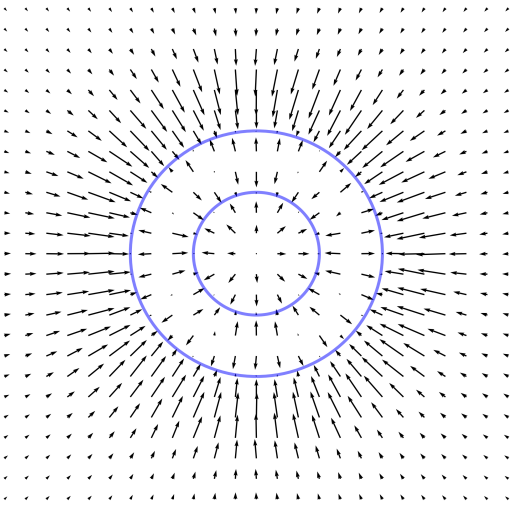
This chapter is adapted from [10]. The author's contributions to this work are as follows:

- **Formulation of Ideas**: I came up with the idea of non-uniform diffusion and how to train non-uniform diffusion models. This idea led to multi-scale diffusion and the CMDE estimator which can be used for training conditional diffusion models. Jan Stanczuk noticed that CMDE is an interpolation between CDE and CdiffE.
- **Experimental Design**: Myself and Jan Stanczuk had equal contribution in the experimental design.

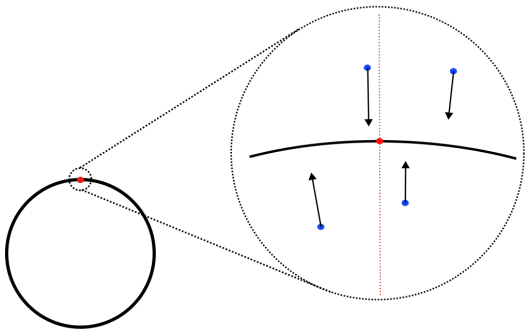
- **Code and Experimental Implementation**: I implemented the experiments presented in the paper. Myself and Jan Stanczuk had equal contribution in the delevopment of the codebase.
- **Theory**: The ideas for the proofs of Theorem 1 and 2 were found together. Final technical ideas were completed by Jan Stanczuk. Jan Stanczuk also compiled the proof of Theorem 3.
- **Presentation**: The design of the paper was discussed together. Jan Stanczuk led the write up of the paper. I contributed to experimental sections of the write up.

This project was conducted under the supervision of Carola-Bibiane Schönlieb and Christian Etmann.

2.3 Diffusion Models Encode the Intrinsic Dimension of Data Manifolds



(a) The data manifold (in blue) and the neural approximation of the score field $\nabla_{\mathbf{x}} \ln p_{t_0}(\mathbf{x})$ obtained from a diffusion model. Near the manifold, the score field is perpendicular to the manifold surface.



(b) The red dot shows a point $\mathbf{x}_0$ on the data manifold where we wish to estimate the dimension. We sample K blue points $\mathbf{x}_t^{(i)}$ in a close neighborhood of the red point and evaluate the score field. The resulting vectors $s_\theta(\mathbf{x}_\varepsilon^{(i)}, \varepsilon)$ point in the normal direction. We put the vectors into a matrix and perform SVD to detect the dimension of the normal space. The dimension of the manifold equals the number of (almost) vanishing singular values.

Fig. 2.5 (Left) Visualization of the score field near the data manifold. (Right) Visualisation of the estimation of the manifold dimension using the trained diffusion model.

In Chapter **??**, we provide a mathematical proof that diffusion models encode data manifolds by approximating their normal bundles. Based on this observation we propose a novel method for extracting the intrinsic dimension of the data manifold from a trained diffusion model. Our insights are based on the fact that a diffusion model approximates the score function i.e. the gradient of the log density of a noise-corrupted version of the target distribution for varying levels of corruption. We prove that as the level of corruption decreases, the score function points towards the manifold, as this direction becomes the direction of maximal likelihood increase. Therefore, at low noise levels, the diffusion model provides us with an approximation of the manifold's normal bundle, allowing for an estimation of the manifold's intrinsic dimension. To the best of our knowledge our method is the first estimator of intrinsic dimension based on diffusion models and it outperforms well established estimators in controlled experiments on both Euclidean and image data.

Originality and Author's Contributions

This chapter is adapted from our published work presented at ICML 2024 on intrinsic dimension estimation with diffusion models [89]. The authors' contributions to this work are as follows:

- **Formulation of Ideas**: Myself and Jan Stanczuk equally contributed to the formulation of ideas.
- **Experimental Design**: Myself and Jan Stanczuk shared equal contributions to the experimental design.
- **Code and Experimental Implementation**: Myself and Jan Stanczuk had equal contribution in the developement of the code framework used to run the experiments. I implemented the following experiments: synthetic image manifolds and MNIST dimensionality estimation. Jan Stanczuk implemented the following experiments: k-spheres, line manifold, spaghetti line, comparison with auto-encoder on MNIST, robustness analysis, and all benchmark methods.
- **Theory**: Jan Stanczuk derived the theoretical results with help from Teo Deveney.
- **Presentation**: Jan Stanczuk and I equally contributed to the write-up, with Teo Deveney contributing to the theoretical sections.

This project was conducted under the supervision of Carola-Bibiane Schönlieb.

2.4 Variational Diffusion Auto-encoder: Latent Space Extraction from Pre-Trained Diffusion Models

Fig. 2.6 Comparison of original and reconstructed images on the FFHQ dataset using our ScoreVAE framework. The left panel presents the original images from the FFHQ dataset, while the right panel displays the corresponding reconstructions generated by ScoreVAE. The results highlight the effectiveness of ScoreVAE in capturing intricate details and preserving high fidelity, overcoming the limitations of traditional VAE models.

In Chapter **??**, we introduce **ScoreVAE**, a novel approach that advances the Variational Autoencoder (VAE) framework by addressing fundamental limitations of conventional VAEs. Traditional VAEs model the reconstruction distribution $p(\mathbf{x}|\mathbf{z})$ as a Gaussian, which often leads to overly smoothed and blurry reconstructions. This limitation arises because the Gaussian assumption fails to capture the complexity and multimodality of real-world data distributions, making it difficult for the model to accurately represent intricate details and sharp features. ScoreVAE addresses this issue by combining a diffusion-time-dependent encoder and an unconditional diffusion model. By employing Bayes' rule for score functions, we analytically derive a robust and flexible model for reconstruction distribution $p(\mathbf{x}|\mathbf{z})$. Our approach bypasses the unrealistic Gaussian assumption, resulting in significantly improved image reconstruction quality.

The ScoreVAE framework also simplifies the training dynamics by decoupling the training of the diffusion model and the encoder. This decoupling enables the use of powerful pre-trained diffusion models that can be readily updated or swapped without retraining the entire system. By separating the prior (diffusion model) from the encoder, ScoreVAE achieves higher fidelity reconstructions compared to traditional VAEs and diffusion decoders.

Our experiments on the CIFAR10 and CelebA datasets demonstrate ScoreVAE's superiority in producing sharper images and reducing reconstruction error. These results underscore the practical advantages of ScoreVAE in handling complex, high-dimensional data, and highlight its potential for improved representation learning and controllable generative modeling.

Originality and Author's Contributions

This chapter is adapted from [9]. The authors' contributions to this work are as follows:

- **Formulation of Ideas**: Myself and Jan Stanczuk contributed equally to the formulation of ideas.
- **Experimental Design**: Myself and Jan Stanczuk had equal contributions to the experimental design.
- **Code and Experimental Implementation**: I implemented the code for ScoreVAE model (encoder-only, corrector), adaptation of discrete prior models to continuous time framework and the following experiments: reconstructions with score-VAE and Diffusion Decoder. Jan Stanczuk implemented the beta-VAE and code for semantic manipulation.
- **Theory**: I suggested the sketch of the proof that entailed connecting the ScoreVAE loss to marginal likelihood [85], and Jan Stanczuk derived the rigorous proof.
- **Presentation**: Myself and Jan Stanczuk contributed equally to the final write-up.

This project was supervised by Carola-Bibiane Schönlieb.

2.5 Score-Based Pullback Riemannian Geometry

In Chapter **??**, we introduce a score-based pullback Riemannian metric that encodes the intrinsic dimensionality and geometry of data under certain distributional assumptions. We show that this data-driven metric can be constructed in practice by modifying normalizing flows with anisotropic base distribution and isometry regularization. This approach yields a scalable framework for computing manifold maps—such as geodesics, exponential maps, distances, and curvature—in closed form. Building on this metric, we additionally construct a Riemannian Auto-encoder (RAE) that recovers the true manifold dimension, offers a global chart of the manifold, and yields an interpretable latent representation thanks to isometry regularization.

(a) Hemisphere (2,3) (b) Sinusoid (1,3) (c) Sinusoid (2,3)

Fig. 2.7 Approximate data manifolds learned by the Riemannian autoencoder generated by score-based pullback Riemannian geometry for three datasets. The orange surfaces represent the manifolds learned by the model, while the blue points correspond to the training data. Each manifold provides a convincing low-dimensional representation of the data, isometric to its respective latent space.

Originality and Author's Contributions

This chapter is adapted from [23]. The authors' contributions to this work are as follows:

- **Formulation of Ideas**: Willem Diepeveen proposed the idea of extracting the data manifold's geometry using a score-based pullback metric. He developed the theoretical motivation of using the score-based metric and showed that the constructed metric allows the computation of manifold maps such as geodesics, exponential map, logarithmic map and distance in closed form. Moreover, he showed that the trained normalizing flow used for the construction of the pullback metric is a Riemannian Auto-encoder.

  I figured out that the proposed metric can be constructed in practice by adapting the framework of Normalizing flows with base distribution anisotropy and l_2 regularisation. Moreover, I discovered the existence of a hessian vector product term that can negatively impact the performance of the Riemannian Auto-encoder if not taken into consideration. Including the hessian vector product term in the regularisation allowed us to use non-affine flows reliably which led to the improvement of the scalability and performance of the method.

- **Experimental Design**: I led the experimental design, incorporating suggestions from Willem Diepeveen and Zakhar Shumaylov.

- **Code and Experimental Implementation**: I implemented the training code and conducted all hyperparameter tuning and experiments. Willem implemented the

functionality for constructing the data-driven manifold maps and the Riemannian autoencoder from a trained model.

- **Theory**: Willem developed the theoretical results. I contributed by explaining the necessity of Hessian-vector product regularization.
- **Presentation**: Zakhar and Willem wrote all sections of the paper except for the experimental section, which I authored.

This project was conducted under the supervision of Carola-Bibiane Schönlieb.

References

[1] Anderson, B. D. (1982). Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326.

[2] Arandjelovic, R. and Zisserman, A. (2017). Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617.

[3] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 214–223. PMLR.

[4] Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Schaeffer, R. D., et al. (2021). Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876.

[5] Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12449–12460.

[6] Bao, H., Dong, L., and Wei, F. (2021). Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*.

[7] Bardes, A., Ponce, J., and LeCun, Y. (2022). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations (ICLR)*.

[8] Batzolis, G., Carioni, M., Etmann, C., Afyouni, S., Kourtzi, Z., and Schönlieb, C.-B. (2024). Caflow: Conditional autoregressive flows. *Foundations of Data Science*, 6(4):553–583. The first author is supported by GSK. Early access: June 2024.

[9] Batzolis, G., Stanczuk, J., and Schönlieb, C.-B. (2023). Variational diffusion auto-encoder: Latent space extraction from pre-trained diffusion models. *arXiv preprint arXiv:2304.12141*.

[10] Batzolis, G., Stanczuk, J., Schönlieb, C.-B., and Etmann, C. (2022). Non-uniform diffusion models.

[11] Behrmann, J., Vicol, P., Wang, K.-C., Grosse, R., and Jacobsen, J.-H. (2021). Understanding and mitigating exploding inverses in invertible neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 1792–1800. PMLR.

[12] Boothby, W. M. (2003). *An introduction to differentiable manifolds and Riemannian geometry, Revised*, volume 120. Gulf Professional Publishing.

[13] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

[14] Butler, K. T., Davies, D. W., Cartwright, H., Isayev, O., and Walsh, A. (2018). Machine learning for molecular and materials science. *Nature*, 559(7715):547–555.

[15] Carmo, M. P. d. (1992). *Riemannian geometry*. Birkhäuser.

[16] Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision (ECCV)*, pages 132–149.

[17] Caron, M., Misra, I., Mairal, J., et al. (2020). Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9912–9924.

[18] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607.

[19] Chen, X. and He, K. (2021). Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*.

[20] Cornish, R., Caterini, A., Deligiannidis, G., and Doucet, A. (2020). Relaxing bijectivity constraints with continuously indexed normalising flows. In *International conference on machine learning*, pages 2133–2143. PMLR.

[21] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019a). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

[22] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019b). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[23] Diepeveen, W., Batzolis, G., Shumaylov, Z., and Schönlieb, C.-B. (2024). Score-based pullback riemannian geometry. *arXiv preprint arXiv:2410.01950*.

[24] Ding, Y., Hu, H., Ge, Z., Wu, J., Zhang, H., and Ding, C. (2021). Vq-vae for multimodal image synthesis. *arXiv preprint arXiv:2103.02398*.

[25] Du, Y. and Mordatch, I. (2019). Implicit generation and modeling with energy-based models. In *Advances in Neural Information Processing Systems*, volume 32.

[26] Esser, P., Rombach, R., and Ommer, B. (2021). Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12873–12883.

[27] Fefferman, C., Mitter, S., and Narayanan, H. (2016). Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049.

[28] Gentile, F., Agrawal, V., Hsing, M.-F., Ton, A.-T., Ban, F., Norinder, U., Gleave, M. E., and Cherkasov, A. (2020). Deep docking: A deep learning platform for ultra-large virtual screening. *Journal of Chemical Information and Modeling*, 60(9):4291–4305.

[29] Gidaris, S., Singh, P., and Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*.

[30] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT Press.

[31] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

[32] Goyal, P., Mahajan, D., Gupta, A., and Misra, I. (2021). Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*.

[33] Grathwohl, W., Wang, K. C., Jacobsen, J.-H., Duvenaud, D., Swersky, K., and Norouzi, M. (2020). Your classifier is secretly an energy-based model and you should treat it like one. In *International Conference on Learning Representations (ICLR)*.

[34] Grill, J.-B., Strub, F., Altché, F., et al. (2020). Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 21271–21284.

[35] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009.

[36] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738.

[37] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). Beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations (ICLR)*.

[38] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

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

[40] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[41] Hossain, M., Sohel, F., Shiratuddin, M. F., and Laga, H. (2020). A comprehensive survey of image caption generation techniques. *Journal of Visual Communication and Image Representation*, 71:102844.

[42] Izmailov, P., Kirichenko, P., Finzi, M., and Wilson, A. G. (2021). Semi-supervised learning with normalizing flows. In *International Conference on Learning Representations*.

[43] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589.

[44] Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410.

[45] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119.

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

[47] Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. (2020). Diffwave: A versatile diffusion model for audio synthesis.

[48] Langevin, P. (1908). Sur la théorie du mouvement brownien. *Comptes Rendus de l'Académie des Sciences (Paris)*, 146:530–533.

[49] LeCun, Y. (2021). Self-supervised learning: The dark matter of intelligence. *Facebook Research*.

[50] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

[51] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[52] LeCun, Y., Chopra, S., and Hadsell, R. (2006). A tutorial on energy-based learning. *Predicting Structured Data*.

[53] Lee, J. M. (2013). Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–31. Springer.

[54] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[55] Lugmayr, A., Danelljan, M., Romero, A., Sabater, N., Timofte, R., and Van Gool, L. (2022). Repaint: Inpainting using denoising diffusion probabilistic models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471.

[56] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *International Conference on Learning Representations (ICLR)*.

[57] Misra, I., Zitnick, C. L., and Hebert, M. (2016). Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision (ECCV)*, pages 527–544. Springer.

[58] Neal, R. M. (2011). MCMC Using Hamiltonian Dynamics. In Brooks, S., Gelman, A., Jones, G. L., and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*, pages 113–162. Chapman and Hall/CRC, Boca Raton, FL.

[59] Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision (ECCV)*, pages 69–84. Springer.

[60] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016a). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

[61] Oord, A. v. d., Kalchbrenner, N., and Kavukcuoglu, K. (2016b). Pixel recurrent neural networks. In *International Conference on Machine Learning (ICML)*, pages 1747–1756.

[62] Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 10256–10265.

[63] Oord, A. v. d., Vinyals, O., and Kavukcuoglu, K. (2017). Neural discrete representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6306–6315.

[64] Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64.

[65] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

[66] Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. (2017). Why and when can deep–but not shallow–networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519.

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

[68] Radford, A., Wu, J., Amodei, D., et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

[69] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr.

[70] Razavi, A., van den Oord, A., and Vinyals, O. (2019). Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*, pages 14866–14876.

[71] Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR.

[72] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, volume 32, pages 1278–1286. PMLR.

[73] Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15).

[74] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

[75] Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326.

[76] Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. (2022). Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[77] Sakai, T. (1996). *Riemannian geometry*, volume 149. American Mathematical Soc.

[78] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, volume 29.

[79] Salimans, T. and Ho, J. (2022). Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*.

[80] Schneider, S., Baevski, A., Collobert, R., Auli, M., and Mohamed, A. (2019). wav2vec: Unsupervised pre-training for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6415–6419.

[81] Scott, D. W. (2015). *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons.

[82] Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall.

[83] Song, J., Meng, C., and Ermon, S. (2020a). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

[84] Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. (2023). Consistency models. *arXiv preprint arXiv:2303.01469*.

[85] Song, Y., Durkan, C., Murray, I., and Ermon, S. (2021a). Maximum likelihood training of score-based diffusion models.

[86] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020b). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

[87] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021b). Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.

[88] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021c). Score-based generative modeling through stochastic differential equations.

[89] Stanczuk, J. P., Batzolis, G., Deveney, T., and Schönlieb, C.-B. (2024). Diffusion models encode the intrinsic dimension of data manifolds. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 46412–46440. PMLR.

[90] Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, J. R., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackermann, Z., et al. (2020). A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.

[91] Sun, C., Myers, A., Vondrick, C., Murphy, K., and Schmid, C. (2019a). Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473.

[92] Sun, H., Mehta, R., Zhou, H. H., Huang, Z., Johnson, S. C., Prabhakaran, V., and Singh, V. (2019b). Dual-glow: Conditional flow-based generative model for modality transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[93] Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Korzhenkov, D., Galyautdinov, I., Kong, N., Goka, H., and Lempitsky, V. (2022). Resolution-robust large mask inpainting with fourier convolutions. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8564–8573.

[94] Tenenbaum, J. B., Silva, V. d., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

[95] Tian, Y. et al. (2022). Understanding self-supervised contrastive learning with generalized contrastive losses. *arXiv preprint arXiv:2202.09671*.

[96] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

[97] Wang, R., Wei, F., Sun, C., Murphy, K., and Schmid, C. (2023). Imagebind: Learning joint representations of vision, audio, and language without supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2256–2266.

[98] Wang, Y., Yang, W., Chen, X., Wang, Y., Guo, L., Chau, L.-P., Liu, Z., Qiao, Y., Kot, A. C., and Wen, B. (2024). Sinsr: Diffusion-based image super-resolution in a single step. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25796–25805.

[99] Wasserman, L. (2006). *All of Nonparametric Statistics*. Springer Science & Business Media.

[100] Xu, H., Ghosh, G., Song, Y., Zhang, X., Li, C.-Y., Sigal, L., and Wang, Y. (2021). Videoclip: Contrastive pre-training for zero-shot video-text understanding. In *Advances in Neural Information Processing Systems*, volume 34, pages 12471–12484.

[101] Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning (ICML)*. PMLR.

[102] Zhang, X., Zhao, P., Zhang, J., Weng, Y., Yuan, H., Li, Y., and Wang, H. (2021). A unified framework for molecule generation and reaction prediction. *Nature Communications*, 12(1):3117.

[103] Zhou, D., Wang, Z., Wang, L., Han, J., Dai, X., Shen, Y., and Feng, J. (2022). ibot: Image bert pre-training with online tokenizer. In *International Conference on Learning Representations*.