Diffusion Models Theory

Akash Malhotra

November 20, 2024

Contents

Chapter 1

Prerequisites

This chapter covers essential concepts that form the foundation for understanding diffusion models: Gaussian distributions, Bayes' theorem, and Kullback-Leibler (KL) divergence. Mastery of these topics will provide a solid base for engaging with diffusion modeling.

1.1 Gaussian Distribution

The Gaussian distribution, also known as the normal distribution, is a continuous probability distribution symmetric around its mean. For a random variable X with mean μ and variance σ^2, it is defined by:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.1}$$

- **Mean (μ)**: Determines the central location of the distribution
- **Variance (σ^2)**: Measures the spread around the mean

The Gaussian distribution has the *highest entropy* among all continuous distributions with a given mean and variance, making it the least biased distribution under these constraints.

1.2 Expectation and LOTUS

For a random variable X with probability density function p(x), its expectation is defined as:

$$\mathbb{E}[X] = \sum_{x \in X} xp(x)dx \tag{1.2}$$

The Law of Unconscious Statistician (LOTUS) states that for any function g(X):

$$\mathbb{E}[g(X)] = \sum_{x \in X} g(x)p(x)dx \tag{1.3}$$

1.3 Bayes' Theorem

Bayes' theorem describes how to update probability estimates given new evidence. For events A (hypothesis) and B (evidence):

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{1.4}$$

where:

- $P(A \mid B)$: Posterior probability
- $P(B \mid A)$: Likelihood
- $P(A)$: Prior probability
- $P(B)$: Marginal probability

This derives from the definition of conditional probability:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \quad \text{and} \quad P(B \mid A) = \frac{P(A \cap B)}{P(A)} \tag{1.5}$$

1.4 KL Divergence

Kullback-Leibler divergence measures the difference between two probability distributions. For distributions P (true) and Q (approximating):

$$D_{KL}(P \parallel Q) = \sum_{x} P(x) \log \frac{P(x)}{Q(x)} \quad \text{(discrete)} \tag{1.6}$$

$$D_{KL}(P \parallel Q) = \int P(x) \log \frac{P(x)}{Q(x)} \, dx \quad \text{(continuous)} \tag{1.7}$$

The KL divergence relates to Shannon entropy $H(P) = -\sum_x P(x) \log P(x)$ through:

$$D_{KL}(P \parallel Q) = -\sum_{x} P(x) \log Q(x) - H(P) \tag{1.8}$$

This shows how much additional uncertainty is introduced when using Q to approximate P. Note that $D_{KL}(P \parallel Q) \geq 0$, with equality if and only if $P = Q$.

Chapter 2

Forward Diffusion Process

The forward diffusion process gradually adds noise to data, motivated by physical diffusion processes. In physics, particles spread from regions of high to low concentration through random motion, reaching maximum entropy. Similarly, in diffusion models, we transform structured data into pure noise through a controlled process.

2.1 Forward Process Definition

Let $\mathbf{x}$ be the original data and $\mathbf{z}_t$ its noisy version at step t. The forward process follows:

$$\mathbf{z}_t = \sqrt{1-\beta_t}\,\mathbf{z}_{t-1} + \sqrt{\beta_t}\,\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \mathbf{I}) \tag{2.1}$$

where $\beta_t \in (0,1)$ controls noise variance and ϵ_t is Gaussian noise. This can be written as:

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}) \tag{2.2}$$

The coefficients serve specific purposes:

- $\sqrt{\beta_t}$: Scales noise to desired variance
- $\sqrt{1-\beta_t}$: Controls information preservation from previous state

2.2 Diffusion Kernel

The joint distribution of all latent variables, given the input, is:

$$q(\mathbf{z}_1, ..., \mathbf{z}_t|\mathbf{x}) = q(\mathbf{z}_1|\mathbf{x}) \prod_{i=2}^{t} q(\mathbf{z}_i|\mathbf{z}_{i-1}) \tag{2.3}$$

Marginalizing over intermediate variables gives the diffusion kernel:

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\mathbf{z}_t|\sqrt{\alpha_t}\mathbf{x}, (1-\alpha_t)\mathbf{I}) \tag{2.4}$$

where:

$$\alpha_t = \prod_{i=1}^{t}(1-\beta_i) \tag{2.5}$$

This can be rewritten as:

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t \tag{2.6}$$

2.3 Key Properties

The forward process has several important characteristics:

- As $T \to \infty$, $q(\mathbf{z}_T|\mathbf{x}) = \mathcal{N}(\mathbf{z}_T|0, \mathbf{I})$
- Each step forms a Markov chain
- All transitions are Gaussian
- The process is fixed rather than learned
- Variance parameters β_t typically follow a schedule where $\beta_1 < \beta_2 < ... < \beta_T$

The marginal distribution of the final state is:

$$q(\mathbf{z}_T) = \mathcal{N}(\mathbf{z}_T|0, \mathbf{I}) \tag{2.7}$$

This process is analogous to an encoder in variational autoencoders but remains fixed during training.

Chapter 3

Reverse Diffusion Process

The reverse diffusion process reconstructs data by gradually removing noise. The goal is to learn conditional distributions $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ that describe how noisy states arise from previous states.

3.1 Reverse Distribution Derivation

We derive $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ by combining known distributions from the forward process. To find the reverse conditional distribution $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$, we apply Bayes' theorem, which states that:

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) = \frac{q(\mathbf{z}_t|\mathbf{z}_{t-1}) \cdot q(\mathbf{z}_{t-1}|\mathbf{x})}{q(\mathbf{z}_t|\mathbf{x})}. \tag{3.1}$$

The conditional distribution $q(\mathbf{z}_t|\mathbf{z}_{t-1})$ represents a Gaussian transition in the forward process, defined as:

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}). \tag{3.2}$$

Next, $q(\mathbf{z}_{t-1}|\mathbf{x})$ specifies how the previous state, $\mathbf{z}_{t-1}$, depends on the original data $\mathbf{x}$:

$$q(\mathbf{z}_{t-1}|\mathbf{x}) = \mathcal{N}(\mathbf{z}_{t-1}|\sqrt{\alpha_{t-1}}\mathbf{x}, (1-\alpha_{t-1})\mathbf{I}). \tag{3.3}$$

Since $q(\mathbf{z}_t|\mathbf{x})$ does not vary with $\mathbf{z_{t-1}}$, we can treat it as a constant term.

Substituting these expressions into Bayes' theorem (3.1) and using the technique "completing the square" yields the desired conditional distribution:

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) = \mathcal{N}(\mathbf{z}_{t-1}|\boldsymbol{m}_t(x, z_t), \sigma_t^2\mathbf{I}), \tag{3.4}$$

where the mean $\boldsymbol{m}_t(x, z_t)$ and variance $\boldsymbol{\sigma}_t^2$ are given by:

$$\boldsymbol{m}_t(x, z_t) = \frac{(1-\alpha_{t-1})\sqrt{1-\beta_t}\mathbf{z}_t + \sqrt{\alpha_{t-1}}\beta_t\mathbf{x}}{1-\alpha_t}, \tag{3.5}$$

$$\boldsymbol{\sigma}_t^2 = \frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}. \tag{3.6}$$

3.2 Neural Network Approximation

Since directly computing $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ is intractable during sampling, as x is unknown, we approximate it with a parameterized model:

$$p(\mathbf{z}_{t-1}|\mathbf{z}_t, w) = \mathcal{N}(\mathbf{z}_{t-1}|\boldsymbol{\mu}(\mathbf{z}_t, w, t), \sigma_t^2\mathbf{I}) \tag{3.7}$$

Here, $\boldsymbol{\mu}$ is predicted by a neural network and σ_t^2 matches the forward process variance.

3.3 Reverse Process Sampling

After training the neural network, the reverse diffusion process can sample new data. Starting from noise $\mathbf{z}_T \sim \mathcal{N}(0, \mathbf{I})$, we iteratively sample from each $p(\mathbf{z}_{t-1}|\mathbf{z}_t, w)$ until we reach $p(\mathbf{z}_0|\mathbf{z}_1)$, approximating the data distribution $p(\mathbf{x})$.

The reverse sampling is given by:

$$p(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_T|w) = p(\mathbf{z}_T) \left\{ \prod_{t=2}^{T} p(\mathbf{z}_{t-1}|\mathbf{z}_t, w) \right\} p(\mathbf{x}|\mathbf{z}_1, \mathbf{w}) \tag{3.8}$$

3.4 Noise Prediction

Instead of predicting denoised images directly, the model can predict the noise component:

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon \tag{3.9}$$

$$\mathbf{x} = \frac{\mathbf{z}_t - \sqrt{1-\alpha_t}\epsilon}{\sqrt{\alpha_t}} \tag{3.10}$$

Substituting this into the reverse mean (3.5) $\mathbf{m}_t(\mathbf{x}, \mathbf{z}_t)$, we get:

$$\mathbf{m}_t(\mathbf{x}, \mathbf{z}_t) = \frac{1}{\sqrt{1-\beta_t}} \left\{ \mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_t \right\} \tag{3.11}$$

Thus, predicting ϵ is equivalent to predicting the denoised image indirectly, as both representations are linearly related. This formulation simplifies training and often leads to better results.

Chapter 4

Training Diffusion Models

The Evidence Lower Bound (ELBO) provides a tractable objective for training diffusion models by maximizing a lower bound on data likelihood.

4.1 ELBO Derivation

The goal is to maximize the log-likelihood of the data under the model:

$$\log p(\mathbf{x}|\mathbf{w}) = \log \int p(\mathbf{x}, \mathbf{z}|\mathbf{w})\, d\mathbf{z}. \tag{4.1}$$

Since direct computation is intractable, we introduce an approximate posterior $q(\mathbf{z}|\mathbf{x})$:

$$\log p(\mathbf{x}|\mathbf{w}) = \log \int q(\mathbf{z}|\mathbf{x}) \frac{p(\mathbf{x}, \mathbf{z}|\mathbf{w})}{q(\mathbf{z}|\mathbf{x})}\, d\mathbf{z}. \tag{4.2}$$

Applying Jensen's inequality, we obtain:

$$\log p(\mathbf{x}|\mathbf{w}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\mathbf{w})}{q(\mathbf{z}|\mathbf{x})} \right]. \tag{4.3}$$

The lower bound is referred to as the Evidence Lower Bound (ELBO):

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log p(\mathbf{x}, \mathbf{z}|\mathbf{w}) - \log q(\mathbf{z}|\mathbf{x}) \right]. \tag{4.4}$$

Maximizing the ELBO corresponds to minimizing the KL divergence between the approximate posterior $q(\mathbf{z}|\mathbf{x})$ and the true posterior $p(\mathbf{z}|\mathbf{x}, \mathbf{w})$.

4.2 ELBO Decomposition

Expanding the joint distribution in $\mathcal{L}(\mathbf{w})$, we get:

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log p(\mathbf{x}|\mathbf{z}, \mathbf{w}) - \mathrm{KL}(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}|\mathbf{w})) \right]. \tag{4.5}$$

The reconstruction term $\log p(\mathbf{x}|\mathbf{z}, \mathbf{w})$ ensures that the generated samples are close to the original data. The KL divergence term regularizes the approximate posterior to align with the prior.

4.3 ELBO Decomposition

Expanding the joint distribution in $\mathcal{L}(\mathbf{w})$, we get:

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z}_1, \mathbf{w}) + \sum_{t=2}^{T} \log \frac{p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})}{q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x})}\right], \tag{4.6}$$

where the reconstruction term $\log p(\mathbf{x}|\mathbf{z}_1, \mathbf{w})$ ensures the generated samples are close to the original data. The consistency terms $\sum_{t=2}^{T} \log \frac{p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})}{q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x})}$ ensure alignment between the forward and reverse processes.

Each consistency term can be rewritten as a KL divergence:

$$\mathrm{KL}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) \parallel p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})), \tag{4.7}$$

where $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ is defined as:

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) = \frac{q(\mathbf{z}_t|\mathbf{z}_{t-1})q(\mathbf{z}_{t-1}|\mathbf{x})}{q(\mathbf{z}_t|\mathbf{x})}. \tag{4.8}$$

Thus, the ELBO becomes:

$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z}_1, \mathbf{w}) - \sum_{t=2}^{T} \mathrm{KL}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) \parallel p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w}))\right], \tag{4.9}$$

where $\mathcal{L}(\mathbf{w})$ is the lower bound. The left term in the bracket is known as reconstruction term and the right term is known as the regularization term that ensures that noise modelled by the neural net is also gaussian noise.

4.4 Network Prediction Approaches

Two approaches for the neural network are:

1. **Predict denoised image**: Network estimates $\mathbf{x}$ from $\mathbf{z}_t$.
2. **Predict noise**: Network $g(\mathbf{z}_t, \mathbf{w}, t)$ estimates ϵ.

Given the forward process:

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon, \tag{4.10}$$

we can recover $\mathbf{x}$:

$$\mathbf{x} = \frac{\mathbf{z}_t - \sqrt{1-\alpha_t}\epsilon}{\sqrt{\alpha_t}}. \tag{4.11}$$

This leads to the reverse mean:

$$\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{z}_t - \sqrt{1-\alpha_t}g(\mathbf{z}_t, \mathbf{w}, t)\right), \tag{4.12}$$

where $\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t)$ is the denoised prediction, and $g(\mathbf{z}_t, \mathbf{w}, t)$ predicts the noise.

4.5 Final Training Objective

To improve training, the network $g(\mathbf{z}_t, \mathbf{w}, t)$ is trained to predict the noise ϵ added at each step, instead of directly predicting the denoised image. This simplifies learning and improves quality.

From the forward process:

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon, \tag{4.13}$$

we can express $\mathbf{x}$ as:

$$\mathbf{x} = \frac{\mathbf{z}_t - \sqrt{1-\alpha_t}\epsilon}{\sqrt{\alpha_t}}. \tag{4.14}$$

The KL divergence between $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ and $p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})$ can be expressed as:

$$\mathrm{KL}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) \parallel p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})) = \frac{\beta_t}{2(1-\alpha_t)(1-\beta_t)} \|g(\mathbf{z}_t, \mathbf{w}, t) - \epsilon_t\|^2 + \text{const.} \tag{4.15}$$

We can also obtain a similar expression for the reconstruction term in (4.9) at time t=1:

$$\ln p(\mathbf{x}|\mathbf{z}_1, \mathbf{w}) = -\frac{1}{2(1-\beta_t)} \|g(\mathbf{z}_1, \mathbf{w}, 1) - \boldsymbol{\epsilon}_1\|^2 + \text{const.} \tag{4.16}$$

In practice, we ignore additive and multiplicative constants. Combining (4.15) and (4.16) and taking all the timesteps into account, we get the following simplified lower bound:

$$\mathcal{L}(\mathbf{w}) = -\sum_{t=1}^{T} \|g(\mathbf{z}_t, \mathbf{w}, t) - \epsilon_t\|^2, \tag{4.17}$$

where $g(\mathbf{z}_t, \mathbf{w}, t)$ estimates the noise ϵ added during the forward process.

Since Lower bound $\mathcal{L}(\mathbf{w})$ (Not a loss function!) maximizes the $\mathbf{p}(\mathbf{x}, \mathbf{w})$, we can remove the negative sign to obtain the loss function, minimizing which would mean maximizing $\mathbf{p}(\mathbf{x}, \mathbf{w})$:

$$Loss(\mathbf{w}) = \sum_{t=1}^{T} \|g(\mathbf{z}_t, \mathbf{w}, t) - \epsilon_t\|^2 \tag{4.18}$$

4.5.1 Interpretation of the Loss Function

For each timestep t, we sample noise ϵ_t to generate $\mathbf{z}_t$. The loss penalizes the difference between the true noise ϵ_t and the network's predicted noise $g(\mathbf{z}_t, \mathbf{w}, t)$, ensuring that the network learns to reverse the diffusion process effectively.

The final objective function that diffusion models minimize is:

$$Loss(\mathbf{w}) = \sum_{t=1}^{T} \|g(\sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t, \mathbf{w}, t) - \epsilon_t\|^2, \tag{4.19}$$

where $g(\sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t, \mathbf{w}, t)$ predicts the noise ϵ_t for the noisy input $\mathbf{z}_t$, providing a stable and efficient training objective for learning the reverse diffusion process.

Chapter 5

DDPM Training and Sampling

5.1 Training and Samping

The complete implementation of diffusion models involves two main algorithms: training the denoising network and generating new samples are given below.

Algorithm 1 Training a denoising diffusion probabilistic model

Require: Training data $\mathcal{D} = \{\mathbf{x}_n\}$
Require: Noise schedule $\{\beta_1, \ldots, \beta_T\}$
Ensure: Network parameters $\mathbf{w}$
  for $t \in \{1, \ldots, T\}$ **do**
    $\alpha_t \leftarrow \prod_{\tau=1}^{t}(1 - \beta_\tau)$ //Calculate alphas from betas
  end for
  repeat
    $\mathbf{x} \sim \mathcal{D}$ //Sample a data point
    $t \sim \{1, \ldots, T\}$ //Sample a point along the Markov chain
    $\epsilon \sim \mathcal{N}(\epsilon|\mathbf{0}, \mathbf{I})$ //Sample a noise vector
    $\mathbf{z}_t \leftarrow \sqrt{\alpha_t}\mathbf{x} + \sqrt{1 - \alpha_t}\epsilon$ //Evaluate noisy latent variable
    $Loss(\mathbf{w}) \leftarrow \|\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t) - \epsilon\|^2$ //Compute loss term
    Take optimizer step
  until converged
  return $\mathbf{w}$ =0

Algorithm 2 Sampling from a denoising diffusion probabilistic model

Require: Trained denoising network $\mathbf{g}(\mathbf{z}, \mathbf{w}, t)$
Require: Noise schedule $\{\beta_1, \ldots, \beta_T\}$
Ensure: Sample vector $\mathbf{x}$ in data space
  $\mathbf{z}_T \sim \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ //Sample from final latent space
  for $t = T, \ldots, 2$ **do**
    $\alpha_t \leftarrow \prod_{\tau=1}^{t}(1 - \beta_\tau)$ //Calculate alpha
    //Evaluate network output
    $\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left\{\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t)\right\}$
    $\epsilon \sim \mathcal{N}(\epsilon|\mathbf{0}, \mathbf{I})$ //Sample a noise vector
    $\mathbf{z}_{t-1} \leftarrow \boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) + \sqrt{\beta_t}\epsilon$ //Add scaled noise
  end for
  $\mathbf{x} = \frac{1}{\sqrt{1-\beta_1}}\left\{\mathbf{z}_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}}\mathbf{g}(\mathbf{z}_1, \mathbf{w}, t)\right\}$ //Final denoising step
  return $\mathbf{x}$ =0

5.2 Implementation Details

The implementation requires:

- Neural network $g(\mathbf{z}, \mathbf{w}, t)$ that predicts noise.
- Noise schedule with $\beta_1 < \beta_2 < ... < \beta_T$.
- Training data and optimization procedure.

Key properties:

- Training uses random timesteps for efficiency.
- Sampling must proceed sequentially.
- Same network parameters used across all timesteps.
- Process is reversible due to small Gaussian transitions.

Chapter 6

Summary

Diffusion models represent a powerful approach to generative modeling that combines simple Gaussian transitions with deep learning. The key aspects of the framework are:

Mathematical Foundations

- Gaussian distributions form the basis of the noise process.
- Bayes' theorem enables reverse process derivation.
- KL divergence guides the training objective.

Core Components

The diffusion framework consists of three main parts:

- **Forward Process**: Gradually adds noise to data following

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon$$

- **Reverse Process**: Learns to denoise using

$$p(\mathbf{z}_{t-1}|\mathbf{z}_t, w) = \mathcal{N}(\mathbf{z}_{t-1}|\boldsymbol{\mu}(\mathbf{z}_t, w, t), \sigma_t^2\mathbf{I})$$

- **Training Objective**: Minimizes noise prediction error

$$\mathcal{L}(\mathbf{w}) = -\sum_{t=1}^{T} \|g(\sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t, \mathbf{w}, t) - \epsilon_t\|^2, \tag{6.1}$$

Key Advantages

- Simple training process without adversarial objectives.
- Stable optimization due to well-behaved Gaussian transitions.
- Flexibility in network architecture and noise schedules.
- Theoretical guarantees from variational inference.

Practical Considerations

- Computational cost scales with number of diffusion steps.
- Trade-off between sample quality and generation speed.
- Memory requirements depend on batch size and model depth.
- Various optimization techniques available for acceleration like DDIM.

Diffusion models have emerged as a powerful framework for generative modeling, offering a principled approach to learning complex data distributions through a combination of simple Gaussian processes and deep neural networks. Their success demonstrates the effectiveness of gradually transforming noise into structured data through learned reverse diffusion.