1141ML-Final Project Report

Reconstructing Ancient Life through Molecular Diffusion Models

分子擴散模型重建古生命

313652018 王宣瑋

September 2025

1 AI 的未來能力：重建古環境

未來二十年，隨著生成式模型特別是擴散模型（Diffusion Model）的演進，AI 可能具備從現存生物分子資料中「推回」過去的能力，重建古代生命的分子結構與蛋白質形態。這種技術不再只是模仿自然，而是嘗試以數據與物理法則爲基礎，生成曾經存在過、但已消逝的生命結構。故若此能力能實現，對人類與科學將具有深遠意義。

首先，它可使我們超越現有化石與古 DNA 的侷限，利用 AI 的推理能力探索「生命從何而來」這一核心問題，重現遠古酵素、細胞機制與代謝系統的可能樣貌。其次，這將爲合成生物學與藥物設計開啓新途徑，AI 所生成的古代或假想蛋白質可能擁有現代生物未具備的穩定性與功能，成爲新材料或新藥設計的重要來源。最後，這樣的突破也將迫使人類重新思考生命的定義與倫理界線——若 AI 能創造一種從未存在但可運作的生命形態，那麼「生命」是否仍僅限於自然演化的產物？

2 Ingredients

- **Data:**
  - ancient DNA fragments to infer plausible ancient molecular states.
  - Simplified toy-model point clouds (e.g., Gaussian mixtures in $\mathbb{R}^3$) to establish core geometric generation abilities.
- **Tools:**
  - Score-based diffusion models for forward noising and iterative reverse denoising.
  - SE(3)-equivariant neural networks (EGNN[1], SE3-Transformer) ensuring that the learned score satisfies
    $$s_\theta(Rx, t) = R\, s_\theta(x, t),$$
    preserving rotational and translational consistency.
  - **Denoising Score Matching (DSM):**
    * DSM provides a fully self-supervised learning signal derived from the diffusion corruption process.
    * The forward diffusion generates noisy samples via
      $$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$
    * The optimal score function satisfies
      $$s^*(x_t, t) = \nabla_{x_t} \log p_t(x_t) = -\frac{\epsilon}{\sigma_t}.$$

* DSM trains the model to approximate this score by minimizing

$$\mathcal{L}_{DSM} = \mathbb{E}_{x_0,\, t,\, \epsilon} \left\| s_\theta(x_t, t) + \frac{\epsilon}{\sigma_t} \right\|^2 .$$

* This avoids computing $\log p(x)$ explicitly and replaces likelihood learning with score learning, which is numerically stable and scalable to high-dimensional 3D structures.
* A simple multilayer perceptron is used as the score network. We aim to mimic the techniques used in EDM[1], RFdiffusion[4], and GEODIFF[5], so we begin with DSM as the foundational step.

3 Machine Learning Paradigm

Primary Paradigm: Unsupervised / Self-supervised Learning.

Why this paradigm?

Because the task of reconstructing ancient molecular structures has no ground-truth labels. The model must learn the underlying data distribution $p(x)$ directly from observed structures, using only the corruption process of diffusion as supervision, rather than human-annotated labels.

Data Source and Target Signal

- **Data source:** 3D molecular or protein structures x, or simplified point clouds in the toy model.
- **Target signal:** The self-supervised denoising objective derived from the forward diffusion process:

$$\frac{\epsilon}{\sigma_t},$$

  which replaces conventional supervised labels and provides a mathematically grounded training signal.

Interaction with the Environment

The model is trained entirely on offline datasets and does not interact with an environment. There is no reward function, no sequential decision-making, and no reinforcement feedback. Thus, the task does *not* involve reinforcement learning.

4. Solvable Model Problem

(1) Problem Design

This toy problem captures the essential abilities required for future 3D molecular diffusion models: denoising, learning $\nabla_x \log p(x)$, and running reverse diffusion.

- **Input:** Noisy samples x_T drawn from $p_T(x)$.
- **Output:** Reconstructed samples x_0' approximating the original distribution $p_0(x)$.
- **Task:** Learn a score model $s_\theta(x)$ such that Langevin dynamics can recover $p_0(x)$.

(2) Model and Method

DSM is adopted as the self-supervised objective:

$$\mathcal{L}_{DSM} = \mathbb{E} \left\| s_\theta(x_t) + \frac{\epsilon}{\sigma} \right\|^2 .$$

The model is trained on synthetic data consisting of two Gaussian clusters or mixtures of Gaussian blobs, chosen because they provide a clean test of distribution recovery.

We use a neural network to solve the toy model. Especially, we use `nn.Linear()` and `nn.SiLU()` to form the layers. For example, in one of our experiments, the `ScoreNet()` class and `dsm_loss()` function are defined as follows:

```
class ScoreNet(nn.Module):
    def __init__(self, x_dim, hidden, num_layers):
        super().__init__()
        layers = [nn.Linear(x_dim, hidden), nn.SiLU()]
        for _ in range(num_layers - 1):
            layers.append(nn.Linear(hidden, hidden))
            layers.append(nn.SiLU())
        layers.append(nn.Linear(hidden, x_dim))
        self.net = nn.Sequential(*layers)

    def forward(self, x):
        return self.net(x)

model = ScoreNet(x_dim=1, hidden=64, num_layers=3).to(device)

def dsm_loss_step(x_dim):
    x_tilde_np, eps_np = sample_noisy_x(batch_size, x_dim, sigma_dsm)
    x_tilde = torch.from_numpy(x_tilde_np).to(device)
    eps     = torch.from_numpy(eps_np).to(device)
    s_hat   = model(x_tilde)
    target  = -eps / sigma_dsm
    loss    = ((s_hat - target)**2).mean()
    return loss
```

For better optimization stability, we employ `optim.Adam(model.parameters(), lr=lr)` as our optimizer, and additionally use `torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer, mode='min', factor=0.5, patience=200, verbose=True)` as a learning rate scheduler.

(3) Experimental Results

We have done three experiments with three dataset:

- Two 1D Gaussian scatters, with mean -2 and $+2$, and variance approximates 1.0. Sample size is $N = 5000$
- Two 2D Gaussian blobs, with mean at $(-5, 0)$ and $(3, 0)$, with both covarient matrix are I_2. Sample size is $N = 5000$
- Mixture of (1) a 2D Gaussian blob at $(-5, 0)$ with covariance I_2 and (2) a noisy linear cluster defined by $y = 0.4x - 5 + \epsilon$, $\epsilon \sim \mathcal{N}(0, 0.2^2)$. Total sample size: $N = 5000$.

Training Curves. The DSM loss consistently decreases and stabilizes around 0.30–0.45 depending on the dataset and noise scale, demonstrating stable learning of the score function.

Forward–Backward Trajectories. By visualizing $x_0 \to x_T \to x_0'$, the trajectories show that the model learns to reverse the noising process, and the histogram of $p_0'(x)$ resembles the original $p_0(x)$.

Distribution Reconstruction. Across different datasets (two Gaussian blobs, blob + noisy linear cluster), the reconstructed distribution $p_0'(x)$ recovers cluster location, shape, and relative density. Langevin samples overlap well with the original training distribution.

(4) Discussion

Through this simplified DSM setting, we gained a clearer understanding of how score matching learns an approximation of $\nabla_x \log p(x)$ and how the reverse process—implemented via Langevin dynamics—can recover high-density regions of the original distribution even when only noisy samples are available. The experiments across different datasets (1D mixtures, 2D Gaussian blobs, and a blob–line mixture) illustrate several important behaviors:

- The DSM loss does not converge to zero but instead stabilizes at a characteristic plateau due to the stochastic nature of the target $-\epsilon/\sigma$.
- The quality of reconstruction strongly depends on the smoothness and accuracy of the learned score function.
- Non-isotropic or non-Gaussian structures (such as the noisy line cluster) are inherently more challenging to recover than simple Gaussian modes.
- Model capacity, the choice of σ, and the number of Langevin steps significantly influence reconstruction fidelity.

Overall, the simplified experiments demonstrate the fundamental "destroy-and-recover" mechanism of diffusion models and highlight how hyperparameter choices shape the effectiveness of the reverse process.

These toy experiments reveal several challenges that will become even more critical when scaling diffusion-based methods to complex, high-dimensional data such as molecular or protein structures:

- **Score estimation in high dimensions.**
- **Trade-off in noise strength.**
- **Limitations of Langevin sampling.**
- **Complex data manifolds.**

These findings highlight both the promise and the limitations of DSM as a building block for more advanced generative models such as GEODIFF, RFdiffusion, and EDM.

A Appendix

Here are the experiment results and figures. And the following are the exact file name with respect to the experiment:

- Two 1D Gaussian scatters: `code-DSM 1D.ipynb`
- Two 2D Gaussian blobs: `code-DSM 2D.ipynb`
- Mixture of a 2D Gaussian blob and a noisy linear cluster: `code-DSM 2D_blobLine.ipynb`

A.1 Two 1D Gaussian scatters

A.1.1 Experiment Settings

- **Dataset**: $x_0 \sim 0.5\mathcal{N}(-2, 1) + 0.5\mathcal{N}(2, 1)$
- **Training** $\sigma = 1.25$; Sample $\sigma = 0.5$. (Here, the σ works for $x_T = x_0 + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$ is noise.)([2])
- **Initial Learning rate**: `lr=5e-3`
- **Runing hardward**: `cpu`

A.1.2 Experiments Results

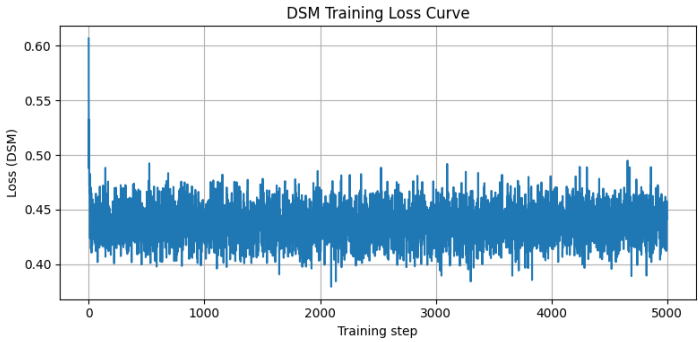

Figure 1: Training loss of the 1D DSM model. After an initial drop, the loss settles into a stable plateau around ~ 0.4, reflecting the inherent stochasticity of the DSM target and confirming that the model has reached a consistent score estimate.

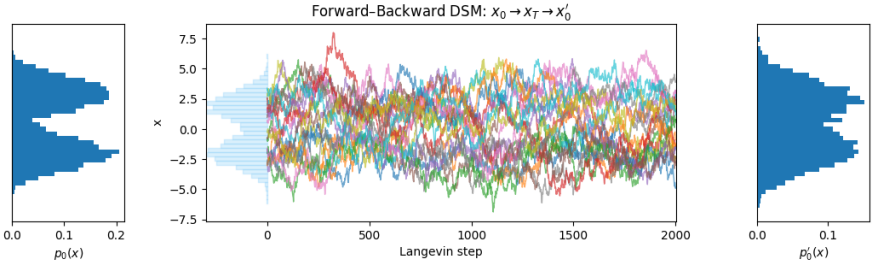

Figure 2: Forward—backward visualization of the DSM process. Samples drawn from the original distribution $p_0(x)$ (left) are corrupted by Gaussian noise to form x_T, and twenty representative noisy samples are tracked through Langevin dynamics (center) as they move back toward high-density regions of the learned score field. The recovered distribution $p'_0(x)$ (right) closely matches the original mixture structure, demonstrating that the model successfully learns a score function capable of reversing the forward noising process.

A.2 Two 2D Gaussian blobs

A.2.1 Experiment Settings

- **Dataset:** A mixture of two 2D Gaussian blobs, $x_0 \sim 0.5\,\mathcal{N}((-5,0), I_2) + 0.5\,\mathcal{N}((3,0), I_2)$.
- **Training noise:** $\sigma_{\text{train}} = 1.25$.
- **Sampling noise:** $\sigma_{\text{sample}} = 0.5$. In both cases, the corrupted data are generated via

$$x_T = x_0 + \sigma\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I_2).$$

- **Initial learning rate:** `lr = 5e-4`.
- **Hardware:** `CPU` (PyTorch backend).

A.2.2 Experiment Results

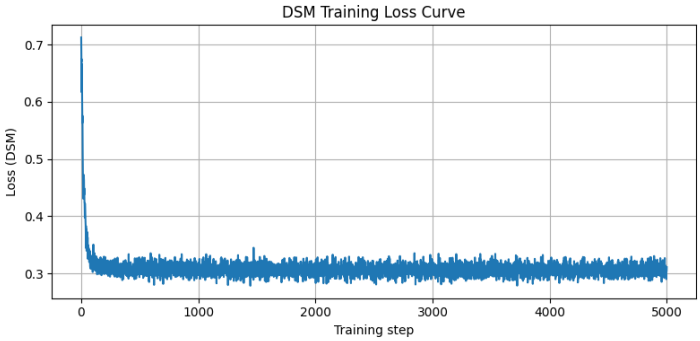

Figure 3: Training loss curve of the DSM model on the 2D two-blob dataset.The loss drops rapidly during the initial optimization phase and stabilizes around 0.30 after approximately 500 steps, indicating that the score network successfully learns a consistent approximation of the noise-induced score target $-\epsilon/\sigma$.The remaining fluctuations are expected due to the stochastic nature of the DSM objective.

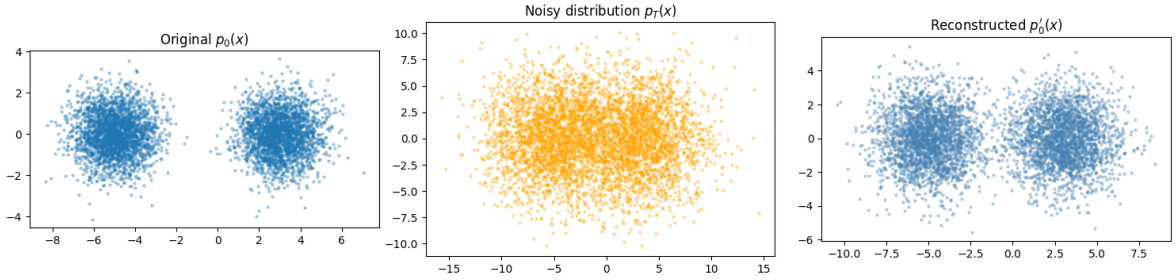

Figure 4: Comparison of the original distribution $p_0(x)$ (left), the noisy distribution $p_T(x)$ obtained by adding Gaussian noise with $\sigma = 0.5$ (middle), and the reconstructed distribution $p'_0(x)$ produced by Langevin dynamics (right). The forward corruption step significantly expands the variance and obscures the two-mode structure, yet the reverse diffusion process successfully recovers both cluster locations and overall geometry. The reconstructed samples exhibit slightly enlarged variance—a known effect of stochastic Langevin updates—but the bimodal structure and density concentration closely match the original distribution, demonstrating that the learned score function effectively inverts the forward noising process.

A.3 Mixture of a 2D Gaussian blob and a noisy linear cluster

A.3.1 Experiment Settings

- **Dataset:** A mixture of
  - a 2D Gaussian blob centered at $(-5, 0)$ with covariance I_2, and
  - a noisy linear cluster defined by

$$y = 0.4x - 5 + \epsilon, \qquad \epsilon \sim \mathcal{N}(0,\, 0.2^2).$$

  A total of $N = 5000$ samples are used.
- **Training noise:** $\sigma_{\text{train}} = 1.20$.
- **Sampling noise:** $\sigma_{\text{sample}} = 0.50$. The noised samples follow

$$x_T = x_0 + \sigma\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I_2).$$

- **Initial learning rate:** `lr = 5e-4`.
- **Hardware:** `CPU` (PyTorch backend).

A.3.2 Experiment Results

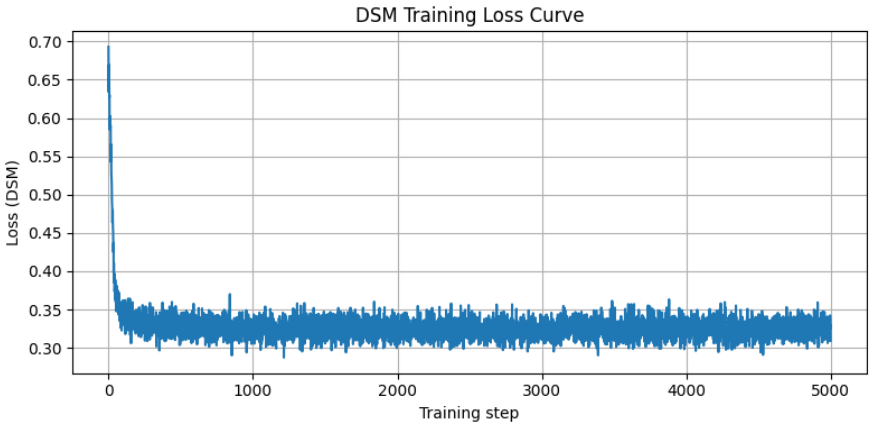

Figure 5: Training loss curve of the DSM model on the 2D mixture dataset. The loss decreases rapidly during the early optimization phase and stabilizes around 0.33–0.35, indicating that the score network has learned a consistent approximation of the target score $-\epsilon/\sigma$. The small oscillations in the later stages are expected due to the stochastic nature of the DSM objective, where each update depends on newly sampled noise.

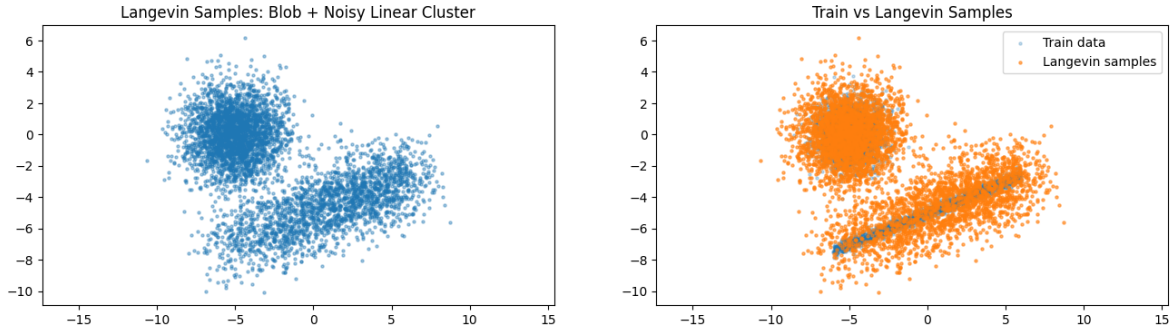

Figure 6: Langevin sampling results on the blob–line mixture dataset. The left panel shows samples generated by running Langevin dynamics starting from Gaussian noise, the right panel compares the generated samples with the training data, demonstrating that the model captures the overall geometry, orientation, and spread of both components. Minor deviations in density concentration are expected due to the stochastic nature of Langevin updates and the complexity of the multimodal data distribution.

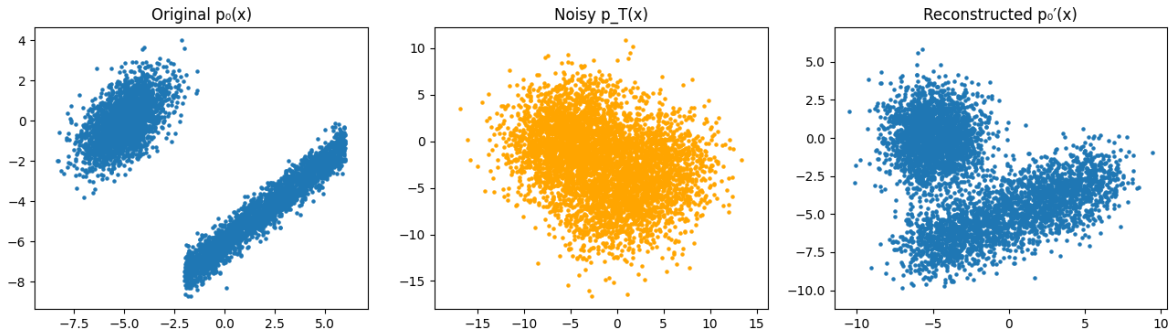

Figure 7: Comparison of the original dataset $p_0(x)$ (left), the noisy distribution $p_T(x)$ obtained by applying Gaussian corruption with $\sigma = 0.5$ (middle)([3]), and the reconstructed distribution $p'_0(x)$ generated by Langevin dynamics (right) on the blob–line mixture dataset. The forward noising step significantly increases variance and blurs both the Gaussian cluster and the linear structure, effectively removing the multimodal geometry. Despite this strong corruption, the reverse diffusion process successfully recovers the two underlying components, capturing both the roughly elliptical blob and the oriented noisy line segment. Mild deformation and increased variance in the reconstruction are expected due to stochastic Langevin sampling and the difficulty of modeling non-isotropic, non-Gaussian structures. Overall, the results demonstrate that the learned score function is capable of inverting the forward process even on heterogeneous data.

References

[1] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pages 8867–8887. PMLR, 2022.

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

[3] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(110):3371–3408, 2010.

[4] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[5] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.