國立臺灣大學 理學院物理學研究所

碩士論文

Department of Physics
College of Science
National Taiwan University
Master's Thesis

通過基於評分的擴散模型實現快速HGCal探測器模擬

Fast HGCal Detector Simulation via Score-Based Diffusion Models

徐振華

Chen-Hua Hsu

指導教授：陳凱風 教授

Advisor: Kai-Feng Chen, Ph.D.

中華民國113年10月

October 2024

NATIONAL TAIWAN UNIVERSITY

MASTER'S THESIS

Fast HGCal Detector Simulation via Score-Based Diffusion Models

Author:
Chen-Hua Hsu

Supervisor:
Dr. Kai-Feng Chen

January 3, 2025

ken91021615@hep1.phys.ntu.edu.tw

Acknowledgements

中文摘要

隨著對撞機的不斷擴建和升級，物理學家面臨著越來越複雜的實驗需求，這導致對計算資源的需求急劇增加。現有的計算能力將難以持續支撐Geant4軟體完成精確且大規模的全套物理計算模擬，因此，尋求一種更加高效、快速的模擬方法已成爲當前的研究重點。在此論文中，我們提出了使用擴散模型作爲核心演算法，並結合transformer模型，嘗試模擬粒子能量在探測器內部的空間分佈。這一方法不僅能夠顯著加速模擬過程，還保持了與Geant4模擬結果相似的精度。本研究的最大特色在於其能夠生成與Geant4預測高度一致的三維能量分佈圖，而不僅僅是如同大多數類似研究所展示的在一維空間上的能量分佈。

關鍵詞：快速模擬、擴散模型、Transformer、CaloChallenge、HGCal。

Abstract

As particle colliders continue to expand and upgrade, physicists face increasingly complex experimental demands, which in turn have led to a sharp rise in the need for computational resources. The current computational power will struggle to support full-scale and precise simulations using Geant4 software, especially as the scale of experiments grows. Therefore, finding a more efficient and fast simulation method has become a pressing priority in current research. In this thesis, we propose using a diffusion model as the core algorithm, coupled with a transformer model, to simulate the spatial distribution of particle energy within the detector. This approach not only significantly accelerates the simulation process but also maintains a level of accuracy comparable to Geant4 simulations. The key feature of this research lies in its ability to generate three-dimensional energy distributions that closely match those predicted by Geant4, rather than the one-dimensional energy distributions typical of most similar studies.

Keywords: Fast Simulation, Diffusion Model, Transformer, CaloChallenge, HGCal.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The upcoming High Luminosity phase of the Large Hadron Collider (LHC) [1] offers unprecedented opportunities to explore new physics in both ATLAS [2] and CMS [3], with its increased luminosity enabling the collection of vast amounts of experimental data. Notably, from Run 2 to Run 3, the luminosity increased by approximately twofold [4].

With a higher collision rate, the collider is expected to produce around 1 billion proton-proton (p-p) collisions per second, captured by detectors containing nearly 100 million readout channels. With only 25 nanoseconds between consecutive groups of colliding protons, new collisions occur even before the previous interactions have fully exited the detector. This massive volume of data not only represents a rich source for scientific discoveries but also poses immense challenges in terms of data processing, storage, and simulation requirements.

Simulation plays a critical role in high-energy physics, as it enables researchers to determine if experimental data aligns with theoretical models. Before delving into deeper analyses, every study must first validate that the observed data is consistent with both background expectations and the signal. This essential step ensures that we have a solid understanding of the main background and signal contributions to each channel, allowing us to apply suitable analysis strategies. However, the high computational demands of this simulation process create bottlenecks, especially as data rates increase. Thus, accelerating the simulation process without compromising accuracy is crucial for timely and reliable analysis.

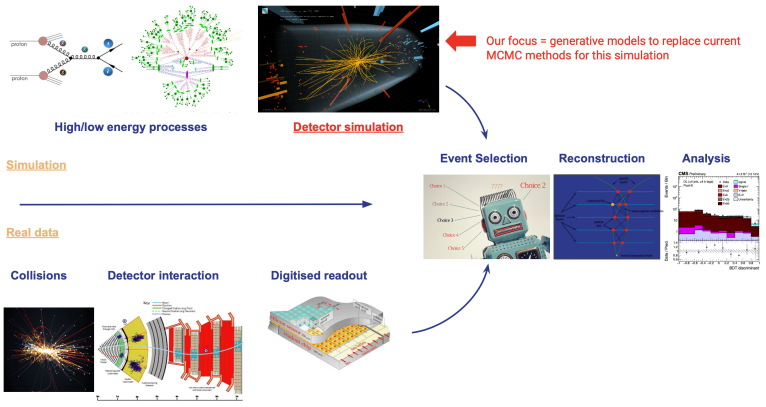


FIGURE 1.1: The importance of simulation. credit: Joshuha Thomas-Wilsker

The simulation of particle interactions and detector responses, traditionally carried out by Monte Carlo methods and implemented with tools like Geant4 [5], has been a foundational aspect of high-energy physics research. However, these methods are computationally intensive and struggle to keep pace with the data rates expected in the Run3 and even future. As the complexity of detector and collider setups continues to grow, so does the time required for full simulations, making it increasingly difficult to scale traditional techniques to meet the demands of modern experiments.

In response to the challenges posed by increasing data rates, generative models—especially diffusion models—have shown promising potential to accelerate simulations without sacrificing quality. Our goal is not to fully replace Geant4 simulations but rather to find a balance between accuracy and speed, as illustrated in Figure 1.2 . Recent works, including Yang et al.'s score-based models [6] and other diffusion approaches in calorimeter simulations [7], have achieved significant reductions in computation time while maintaining fidelity. Building on these innovations, our project introduces a novel model designed to generate 3D point clouds representing energy distributions across spatial coordinates in one step. Unlike previous models, which often focus on single-dimensional energy profiles (e.g., energy vs. z-coordinate), our approach captures the full 3D energy distribution in a single forward pass, allowing for rapid and comprehensive simulations that could match the data collection demands of high-luminosity experiments.

Detailed detector simulations are crucial in particle and nuclear physics data analysis. They allow researchers to compare particle-level predictions with observed data and to account for detector effects, making it possible to accurately interpret experimental results and compare them with theoretical predictions. Simulations also play a significant role in designing future experiments, guiding adjustments to optimize detector performance [8, 9]. Geant4-based simulations [10] have become a standard

tool in high-energy physics due to their precision, but achieving this precision is computationally demanding. Particle propagation in dense materials produces numerous secondary particles that undergo complex electromagnetic and nuclear interactions, making calorimeters—whose role is to measure deposited energy—the most challenging detectors to simulate accurately. In fact, a large portion of computing resources in high-energy physics is dedicated to simulating particle propagation in dense materials using Geant4.

The experiments at the Large Hadron Collider (LHC) generate billions of events per run, each with hundreds to thousands of individual calorimeter showers. Due to computing budget constraints, it is not feasible to use Geant4 simulations for all events, so experiments have developed fast simulation methods. These methods replace physics-based models with simpler parametric models calibrated to full simulations. While these fast simulations are efficient, their simplified parameterizations limit their accuracy, especially for modeling complex, high-dimensional correlations. Often, only a few one-dimensional observables are optimized, which may not fully capture the intricacies of particle interactions.

Deep learning provides a compelling alternative to traditional parametric models, with generative approaches like Generative Adversarial Networks (GANs) [11], Variational Autoencoders (VAEs) [12], and Normalizing Flows (NFs) [13] increasingly adapted for fast detector simulations. GANs, for example, have demonstrated considerable speed and adaptability in generating calorimeter showers [14] and are now even integrated into the ATLAS experiment's fast simulation framework [15]. Nonetheless, GANs present optimization challenges and can suffer from mode collapse,where the generator produces a narrow range of outputs, failing to capture the diversity of the data distribution. Conversely, NFs provide robust training and accurate density estimation, yet they remain computationally demanding when applied to high-dimensional data, which limits their practicality for simulating complex detector responses [16, 17].

In this work, we explore score-based generative models [6], which learn the gradient of the data density rather than the density itself, enabling a more flexible network architecture without requiring the Jacobian computation during training. This flexibility supports the use of bottleneck layers, reducing trainable parameters and improving scalability. Recent advances in score-based generative models have shown potential in calorimeter simulation, achieving a balance between high-dimensional fidelity and computational efficiency, making them suitable for ultra-fine calorimeters and other high-complexity datasets [18, 19].

By leveraging score-based models, our project aims to address both the demands of

the high-luminosity phase of the LHC and the limitations of traditional fast simulation methods. Our approach enhances accuracy by capturing full 3D spatial distributions while significantly reducing the time required for simulation, thus providing a scalable, reliable solution for next-generation collider experiments.

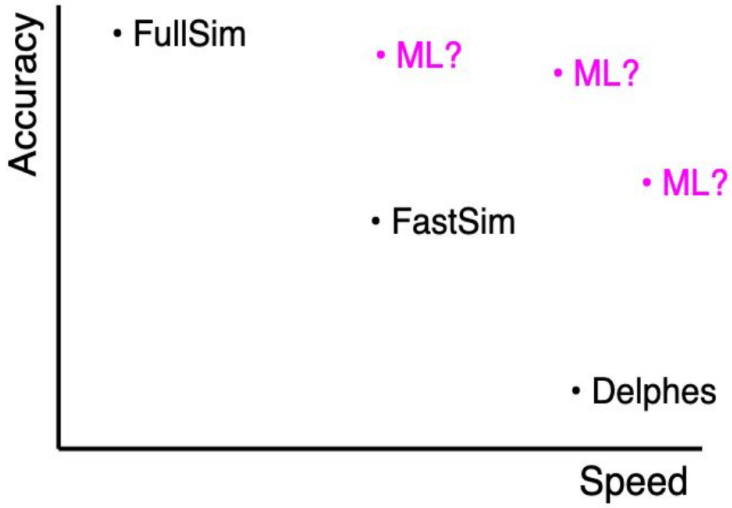


FIGURE 1.2: The balance between accuracy and speed in simulation. credit: Joshuha Thomas-Wilsker

1.2 Challenges

The generation of a 3D point cloud to depict energy deposition across spatial coordinates introduces unique challenges. Existing approaches primarily model the relationship between energy and a single spatial dimension, typically generating only partial representations of energy distributions. Our model, by contrast, aims to capture the complete three-dimensional energy profile in a single forward pass, which requires balancing high-dimensional fidelity and computational efficiency.

To achieve this goal, our model leverages advanced features, including Gaussian Fourier Projection for time encoding and mean-field attention mechanisms with a class token, in addition to conditional guidance based on incident energy. These architectural choices allow us to control both positional and energy distributions, addressing the intricacies of accurate 3D spatial modeling. However, managing the computational load and also let model learn every relation between each variables is quite hard.

This high-dimensional generative task requires careful conditioning to reflect realistic variations in energy deposition across multiple spatial coordinates, especially

given the model's need to dynamically adjust based on the incident energy. Achieving this balance involves a tradeoff between accuracy and computational load, as the high fidelity demanded in multi-dimensional output often requires extensive computation. Nevertheless, our optimized approach achieves up to a 100-fold speedup over traditional simulation methods, providing a scalable solution that addresses the needs of next-generation collider experiments.

In summary, our project seeks to address both the demands of high luminosity and the constraints of traditional simulations, aiming to bridge the gap between scalability and fidelity in particle shower simulations. Our advancements in 3D point cloud generation not only enhance the efficiency of simulations but also mark a step forward in producing realistic, high-dimensional data essential for future discoveries in high-energy physics.

Chapter 2

Detector

Chapter 3

Dataset

3.1 Geant4 Simulation

3.2 CaloChallenge

Dataset1 Dataset2

3.3 Data Preprocessing

Bucketing
Preprocessor The reason of choosing x y coordinate rather than spherical coordinate

3.4 The Fast Calorimeter Simulation Challenge (CaloChallenge)

The Fast Calorimeter Simulation Challenge, or CaloChallenge, is an initiative designed to advance the development of fast, accurate, and efficient generative models for calorimeter shower simulations. This challenge bridges the gap between traditional simulation methods like GEANT4 and novel machine learning approaches, providing datasets, benchmarks, and metrics for evaluation [**calochallenge**].

3.4.1 Objectives

CaloChallenge has the following primary goals:

- Encourage the development of generative models capable of fast and accurate calorimeter shower simulation.
- Provide standardized datasets and metrics for consistent evaluation and benchmarking.
- Foster collaboration across the high-energy physics and machine learning communities.

3.4.2 Datasets

The CaloChallenge offers three distinct datasets, each increasing in complexity, to evaluate model performance in diverse scenarios. The datasets are as follows:

Dataset 1: Single-Layer Geometry

Dataset 1 features a simplified calorimeter geometry with a single cylindrical layer. The layer is segmented radially into 51 bins. This dataset is designed for testing the ability of generative models to simulate energy deposition in a straightforward configuration. It provides a foundation for understanding basic model capabilities before progressing to more complex geometries.

Dataset 2: Multi-Layer Geometry

Dataset 2 introduces a more complex geometry with a 3D arrangement of 45 cylindrical layers. Each layer is divided into 5 radial bins and 16 angular bins, creating a voxelized representation of the energy deposition. This dataset challenges models to handle both radial and angular dependencies, making it a significant step up from Dataset 1.

Dataset 3: Realistic Calorimeter Geometry

Dataset 3 simulates a realistic high-granularity calorimeter with 45 cylindrical layers, 5 radial bins, and 16 angular bins, but with additional complexities in the detector geometry and particle distributions. It is designed to evaluate a model's ability to generalize and simulate realistic scenarios encountered in particle physics experiments.

3.4.3 Data Format

Each dataset is stored as one or more HDF5 files created using Python's `h5py` module with gzip compression. The files include:

- `incident_energies`: An array of shape (`num_events`, 1) containing the incoming particle energies in MeV.
- `showers`: An array of shape (`num_events`, `num_voxels`) storing the energy depositions (in MeV) for each voxel, flattened in a specific order.

The mapping of voxel indices to spatial coordinates follows the detector segmentation. Helper functions are provided for reshaping and handling the data.

3.4.4 Evaluation Metrics

CaloChallenge evaluates the generative models using multiple metrics, including:

- A binary classifier trained to distinguish between real GEANT4 samples and model-generated samples.
- Chi-squared comparisons between histograms of high-level features, such as layer energies and shower shapes.
- Speed and resource usage metrics, such as training time, generation time, and memory footprint.
- Interpolation capabilities to test generalization across unseen particle energies.

3.4.5 Community Engagement

Participants are encouraged to share their findings and contribute to community discussions. The challenge concludes with a workshop to present results, compare approaches, and collaborate on a community paper documenting the outcomes. For communication and updates, participants can join the ML4Jets Slack channel and the Google Groups mailing list [**calochallenge**].

For further details, visit the official CaloChallenge GitHub repository: https://github.com/CaloChallenge/homepage.

Chapter 4

Algorithm

4.1 AE

4.2 VAE

test

4.3 Score-based Diffusion Model

One drawback of Variational Autoencoders (VAEs) is the inclusion of the KL divergence term in their loss function. While VAEs are effective for compressing data (encoding), they struggle to generate high-quality, diverse samples. This limitation stems from their reliance on sampling from a normal distribution in the latent space. Although VAEs are trained to bring the posterior distribution close to a Gaussian, in practice, the match is often not precise enough to ensure that samples drawn from this distribution will be of high quality.

Therefore, an alternative approach, introduced in 2015, is the “diffusion model,” which can be implemented using either score-matching or denoising techniques. Diffusion models aim to generate synthetic data based on a set of independent, identically distributed (i.i.d.) samples drawn from an unknown data distribution. The key concept is to simulate new samples by either employing denoising Score Langevin Dynamics (SMLD) or implementing Denoising Diffusion Probabilistic Model (DDPM), where a deep neural network approximates the score, or gradient, of the log-density of the data distribution. Next we will discuss the two methods in detail.

4.3.1 Denoising Score Matching with Langevin Dynamics (SMLD)

Langevin Dynamics in generative modeling is a way to generate samples by simulating a process that gradually moves from random points in space toward areas with high probability density, where most of the real data is located. It does this by changing along the directions defined by the gradient of the probability distribution, called the "score" in our context. At each step, a small amount of Gaussian noise is added

to introduce randomness, ensuring that each path taken is unique and prevents the sampling process from getting "stuck" in local regions.

In simpler terms, think of Langevin Dynamics as a guided walk starting from a random spot and following a path that gradually leads toward more typical or likely values of the data (like images, text, etc.). The direction of each step is influenced by both the data structure (moving toward areas where data is dense) and a bit of noise to keep things varied, which helps to explore the whole space more effectively. This makes Langevin Dynamics an effective sampling method for creating new data points in generative modeling.

In this approach, we define a perturbation mechanism $p_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}; x, \sigma^2 I)$, which acts as a Gaussian kernel centered at x with variance σ^2. This perturbation is integrated over the data distribution $p_{\text{data}}(x)$ to yield the broader distribution $p_\sigma(\tilde{x}) = \int p_{\text{data}}(x) p_\sigma(\tilde{x}|x)\, dx$.

We consider a range of increasing noise scales, where $\sigma_{\min} = \sigma_1 < \sigma_2 < \cdots < \sigma_N = \sigma_{\max}$. Typically, $\sigma_{\min}$ is chosen to be small enough that $p_{\sigma_{\min}}(x) \approx p_{\text{data}}(x)$, capturing the original data distribution, while $\sigma_{\max}$ is set large enough so that $p_{\sigma_{\max}}(x) \approx \mathcal{N}(x; 0, \sigma_{\max}^2 I)$, resembling a Gaussian prior.

Following the work of Song and Ermon [20], we train a Noise Conditional Score Network (NCSN), denoted $s_\theta(x, \sigma)$, by minimizing a weighted sum of denoising score matching objectives as follows:

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} \sigma_i^2 \, \mathbb{E}_{p_{\text{data}}(x)} \, \mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \left[\| s_\theta(\tilde{x}, \sigma_i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x}|x) \|_2^2 \right]. \tag{4.1}$$

Given sufficient data and model capacity, the resulting score-based model $s_\theta^*(x, \sigma)$ estimates the gradient $\nabla_x \log p_\sigma(x)$ across noise scales $\sigma \in \{\sigma_i\}_{i=1}^N$.

So the Langevin Dynamics process can be described as follows:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\mathbf{z}_{i-1}, i = 1, 2, \ldots, N \tag{4.2}$$

4.3.2 Denoising Diffusion Probabilistic Model (DDPM)

Next, we are going to introduce the second method for diffusing models, the Denoising Diffusion Probabilistic Model (DDPM) [21]. Unlike SMLD, DDPM incorporates a scaling factor for x, which modifies the approach slightly. The basic idea is to define the conditional probability distribution as follows: $p(x_i|x_{i-1}) = \mathcal{N}\left(x_i; \sqrt{1-\beta_i} x_{i-1}, \beta_i I\right)$.

Following Sohl-Dickstein et al. (2015) and Ho et al. (2020), let us consider a set of positive noise scales $0 < \beta_1, \beta_2, \ldots, \beta_N < 1$. For each data point $x_0 \sim p_{\text{data}}(x)$, we define a discrete Markov chain $\{x_0, x_1, \ldots, x_N\}$, with each transition given by $p(x_i|x_{i-1}) = \mathcal{N}\left(x_i; \sqrt{1-\beta_i}x_{i-1}, \beta_i I\right)$. Consequently, we can write the marginal distribution $p_{\alpha_i}(x_i|x_0) = \mathcal{N}\left(x_i; \sqrt{\alpha_i}x_0, (1-\alpha_i)I\right)$, where $\alpha_i := \prod_{j=1}^{i}(1-\beta_j)$.

As in SMLD, we also train it by minimizing the denoising score matching objective:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} (1-\alpha_i)\, \mathbb{E}_{p_{\text{data}}(x)}\, \mathbb{E}_{p_{\alpha_i}(\tilde{x}|x)} \left[\| s_\theta(\tilde{x}, \alpha_i) - \nabla_{\tilde{x}} \log p_{\alpha_i}(\tilde{x}|x) \|_2^2 \right]. \tag{4.3}$$

where again, $1-\alpha_i$ is just a weighting factor.

What's more, we can define the perturbed data distribution as $p_{\alpha_i}(\tilde{x}) := \int p_{\text{data}}(x) p_{\alpha_i}(\tilde{x}|x) dx$. The noise scales are chosen so that x_N approximates a standard normal distribution $\mathcal{N}(0, I)$. So the simialr form as SMLD will be

$$x_{t-1} = \sqrt{1-\beta_t}x_t + \sqrt{\beta_t}z_t, \quad t = N, N-1, \ldots, 1 \tag{4.4}$$

where $z_t \sim \mathcal{N}(0, I)$ are standard normal samples. The final sample x_0 is drawn from the data distribution $p_{\text{data}}(x)$. The process is repeated for each data point, and the final samples are generated by running the Markov chain for T steps. The resulting samples are expected to approximate the data distribution $p_{\text{data}}(x)$ when $T \to \infty$ under suitable conditions.

4.4 Forward Process

So far, we have discussed two ways of simulating new samples from a given data distribution. Although they look different, both methods are based on the same principle: iteratively transforming a sample from a simple distribution (e.g., a Gaussian) to a more complex one (e.g., the data distribution).

Based on the work of Yang Song [20], we can generalize this concept through what is called the **forward process** in diffusion models.

Our goal is to construct a diffusion process x_t indexed by a continuous time variable $t \in [0, T]$, such that:

$$x_0 \sim p_0 \tag{4.5}$$

for which we have a dataset of independent and identically distributed (i.i.d.) samples, and

$$x_T \sim p_T \tag{4.6}$$

for which we have a tractable form to generate samples efficiently. In other words, p_0 is the data distribution and p_T is the prior distribution.

This diffusion process can be modeled as the solution to an Itô stochastic differential equation (SDE):

$$dx = f(x,t)\,dt + g(t)\,dw \tag{4.7}$$

where:

- x is the state variable,
- $f(x,t)$ is the drift coefficient,
- $g(t)$ is the diffusion coefficient,
- w is a Wiener process (Brownian motion).

For later we can show that this has a slightly better result than orginal DDPM and SMLD.

4.5 Backward Process

With the forward process established, we can now construct the **backward process**. The aim of this process is to generate samples from the data distribution p_0, given samples from the prior distribution p_T.

The continuous form of this process is defined by the following stochastic differential equation (SDE):

$$d\mathbf{x} = \mathbf{f}_t(\mathbf{x})\,dt + g_t\,d\mathbf{w} \tag{4.8}$$

To direclty prove the reverse SDE formula in continuous form will be a little complex. But we can get the same spirit from discrete form, as $\Delta t \to 0$, the continuous equation above can be approximated by:

$$\mathbf{x}_{t+\Delta t} - \mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_t)\,\Delta t + g_t\sqrt{\Delta t}\,\varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{4.9}$$

The discrete form of the stochastic differential equation (SDE) is especially valuable for practical computer implementations. By breaking down the continuous process into discrete steps, we can simulate both the diffusion and reverse processes incrementally, allowing us to generate samples using numerical methods. This approach enables us to approximate continuous dynamics with a series of discrete updates, making the computations more manageable and efficient.

In this way, using the SDE framework to describe diffusion models provides a clear distinction between theoretical analysis and practical implementation. We can rely on the mathematical properties of continuous SDEs for analysis, while in actual applications, we have the flexibility to choose any appropriate discretization method for efficient numerical calculation.

In probabilistic terms, Equation (4.9) implies that the conditional probability is given by

$$\begin{aligned} p(\mathbf{x}_{t+\Delta t}|\mathbf{x}_t) &= \mathcal{N}\left(\mathbf{x}_{t+\Delta t}; \mathbf{x}_t + \mathbf{f}_t(\mathbf{x}_t)\,\Delta t, g_t^2\,\Delta t\,\mathbf{I}\right) \\ &\propto \exp\left(-\frac{\|\mathbf{x}_{t+\Delta t} - \mathbf{x}_t - \mathbf{f}_t(\mathbf{x}_t)\,\Delta t\|^2}{2g_t^2\Delta t}\right) \end{aligned} \tag{4.10}$$

Now since our goal is to use the forward process to derive the backward process, which means obtaining $p(\mathbf{x}_t|\mathbf{x}_{t+\Delta t})$, we apply Bayes' theorem, as shown in "A Discussion on Generative Diffusion Models: DDPM = Bayesian + Denoising":

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{x}_{t+\Delta t}) &= \frac{p(\mathbf{x}_{t+\Delta t}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{x}_{t+\Delta t})} \\ &= p(\mathbf{x}_{t+\Delta t}|\mathbf{x}_t)\exp\left(\log p(\mathbf{x}_t) - \log p(\mathbf{x}_{t+\Delta t})\right) \\ &\propto \exp\left(-\frac{\|\mathbf{x}_{t+\Delta t} - \mathbf{x}_t - \mathbf{f}_t(\mathbf{x}_t)\Delta t\|^2}{2g_t^2\Delta t} + \log p(\mathbf{x}_t) - \log p(\mathbf{x}_{t+\Delta t})\right) \end{aligned} \tag{4.11}$$

It is not difficult to see that when Δt is sufficiently small, $p(\mathbf{x}_{t+\Delta t}|\mathbf{x}_t)$ will be significantly non-zero only when $\mathbf{x}_{t+\Delta t}$ is close to $\mathbf{x}_t$. Conversely, only in this case will $p(\mathbf{x}_t|\mathbf{x}_{t+\Delta t})$ be significantly non-zero. Therefore, we only need to conduct an approximate analysis for situations where $\mathbf{x}_{t+\Delta t}$ is close to $\mathbf{x}_t$. For this, we can use a Taylor expansion:

$$\log p(\mathbf{x}_{t+\Delta t}) \approx \log p(\mathbf{x}_t) + (\mathbf{x}_{t+\Delta t} - \mathbf{x}_t)\cdot\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t) + \Delta t\,\frac{\partial}{\partial t}\log p(\mathbf{x}_t) \tag{4.12}$$

It is important not to neglect the term $\frac{\partial}{\partial t}$, because $p(\mathbf{x}_t)$ is a function of both t and $\mathbf{x}_t$, while $p(\mathbf{x}_{t+\Delta t})$ is a function of $t+\Delta t$ and $\mathbf{x}_{t+\Delta t}$. Thus, $p(\mathbf{x}_t)$ must include an additional time derivative. Substituting this into Equation (4.11) allows us to derive:

$$p(\mathbf{x}_t|\mathbf{x}_{t+\Delta t}) \propto \exp\left(-\frac{\|\mathbf{x}_{t+\Delta t} - \mathbf{x}_t - \left[\mathbf{f}_t(\mathbf{x}_t) - g_t^2\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t)\right]\Delta t\|^2}{2g_t^2\Delta t} + \mathcal{O}(\Delta t)\right) \tag{4.13}$$

As $\Delta t \to 0$, the term $\mathcal{O}(\Delta t)$ becomes negligible, thus:

$$p(\mathbf{x}_t|\mathbf{x}_{t+\Delta t}) \propto \exp\left(-\frac{\|\mathbf{x}_{t+\Delta t} - \mathbf{x}_t - \left[\mathbf{f}_t(\mathbf{x}_t) - g_t^2 \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)\right] \Delta t\|^2}{2g_t^2 \Delta t}\right) \tag{4.14}$$

Finally, the above formula indicates that the reverse process also contains a deterministic part and a stochastic part. The deterministic part consists of $\mathbf{f}_t(\mathbf{x}_t) - g_t^2 \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$, while the stochastic part is $g_t\sqrt{\Delta t}\varepsilon$.

Thus, our reverse process is defined as:

$$\mathbf{x}_{t-\Delta t} = \mathbf{x}_t - \left[\mathbf{f}_t(\mathbf{x}_t) - g_t^2 \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)\right] \Delta t + g_t\sqrt{\Delta t}\varepsilon \tag{4.15}$$

We can use the picture below to illustrate the forward and backward processes in a diffusion model:

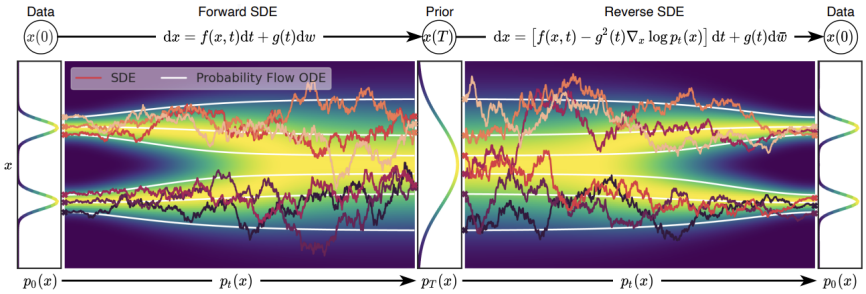


FIGURE 4.1: Forward and Backward Processes in Diffusion Models (The picture is from Song and Ermon (2019))

4.6 VE, VP SDEs

4.6.1 Continuos Forward Process

After we established the general form of the forward and backward processes, we can now go back to see how to apply them on SMLD (VE mthoed) and DDPM (VP method).

So in this section, we try to present detailed derivations demonstrating that the noise perturbations in SMLD (Score-based generative modeling via Langevin Dynamics) and DDPM (Denoising Diffusion Probabilistic Models) are discretizations of the Variance Exploding (VE) and Variance Preserving (VP) Stochastic Differential Equations (SDEs), respectively. We also introduce sub-VP SDEs, which are modifications of VP SDEs that often yield improved performance in terms of sample quality and likelihoods.

First, when utilizing a total of N noise scales, each perturbation kernel $p_{\sigma_i}(x|x_0)$ for SMLD can be derived from the following Markov chain:

$$x_i = x_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} z_{i-1}, \quad i = 1, 2, \ldots, N, \tag{4.16}$$

where $z_{i-1} \sim \mathcal{N}(0, I)$ and $x_0 \sim p_{\text{data}}$. Here, we introduce $\sigma_0 = 0$ for simplicity. As $N \to \infty$, the Markov chain $\{x_i\}_{i=1}^N$ converges to a continuous stochastic process $\{x(t)\}_{t=0}^1$, and $\{\sigma_i\}_{i=1}^N$ becomes a function $\sigma(t)$, while z_i transitions to $z(t)$. We denote the continuous time variable as $t \in [0, 1]$ instead of the integer index $i \in \{1, 2, \ldots, N\}$. Let $\mathbf{x}(\frac{i}{N}) = \mathbf{x}_i$, $\sigma(\frac{i}{N}) = \sigma_i$, $\mathbf{z}(\frac{i}{N}) = \mathbf{z}_i$, for $i = 1, 2, \ldots, N$. Rewriting Equation (20) with $\Delta t = \frac{1}{N}$ gives:

$$x(t + \Delta t) = x(t) + \sqrt{\sigma^2(t + \Delta t) - \sigma^2(t)} z(t) \approx x(t) + \sqrt{\frac{d\sigma^2(t)}{dt}} \Delta t z(t), \tag{4.17}$$

where the approximation holds as $\Delta t \to 0$. In the limit of $\Delta t \to 0$, we obtain the VE SDE:

$$dx = \sqrt{\frac{d\sigma^2(t)}{dt}} dw. \tag{4.18}$$

Furthermore, we usually let σ sequence to be a geometric sequence. We have $\sigma(\frac{i}{N}) = \sigma_i = \sigma_{\min}(\frac{\sigma_{\max}}{\sigma_{\min}})^{\frac{i-1}{N-1}}$ for i ranges from 1 to N. If $N \to \infty$

The corresponding VE SDE is

$$d\mathbf{x} = \sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}} \right)^t \sqrt{2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}} \right)} d\mathbf{w}, \quad t \in (0, 1). \tag{4.19}$$

For the perturbation kernels $p_{\alpha_i}(x|x_0)$ used in DDPM, the discrete Markov chain is given by:

$$\mathbf{x}_i = \sqrt{1 - \beta_i} \mathbf{x}_{i-1} + \sqrt{1 - \beta_i} z_{i-1}, \quad i = 1, 2, \ldots, N, \tag{4.20}$$

where $z_{i-1} \sim \mathcal{N}(0, I)$. To obtain the limit of this Markov chain as $N \to \infty$, we define an auxiliary set of noise scales $\{\bar{\beta}_i\}_{i=1}^N$ and rewrite Equation (22) as follows:

$$\mathbf{x}_i = \sqrt{1 - \bar{\beta}_i} \mathbf{x}_{i-1} + \sqrt{1 - \bar{\beta}_i} z_{i-1}, \quad i = 1, 2, \ldots, N, \tag{4.21}$$

As $N \to \infty$, the noise scales $\{\bar{\beta}_i\}_{i=1}^N$ converge to a function $\beta(t)$ indexed by $t \in [0, 1]$. Let $\{\bar{\beta}_i\}_N = \beta$ and $\{x_i\}_N = x$ and $\{z_i\}_N = z$. Rewriting Equation (23) gives:

$$\begin{aligned}\mathbf{x}(t+\Delta t) &= \sqrt{1-\beta(t+\Delta t)}\mathbf{x}(t) + \sqrt{1-\beta(t+\Delta t)}z(t) \\ &\approx \mathbf{x}(t) - \frac{1}{2}\beta(t+\Delta t)\Delta t\mathbf{x}(t) + \sqrt{\beta(t+\Delta t)\Delta t}\mathbf{z}(t) \\ &\approx \mathbf{x}(t) - \frac{1}{2}\beta(t)\Delta t\mathbf{x}(t) + \sqrt{\beta(t)\Delta t}\mathbf{z}(t)\end{aligned} \tag{4.22}$$

where the approximation holds as $\Delta t \to 0$. Therefore, in the limit of $\Delta t \to 0$, we obtain the VP SDE:

$$d\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,dt + \sqrt{\beta(t)}\,d\mathbf{w}. \tag{4.23}$$

As in DDPM, β is typically an arithmetic sequence where $\beta_i = \beta_{\min} + t(\beta_{\max} - \beta_{\min})$ for t ranges from 0 to 1 if $N \to \infty$. This will then give us the VP SDE as

$$d\mathbf{x} = -\frac{1}{2}\left(\beta_{\min} + t(\beta_{\max} - \beta_{\min})\right)\mathbf{x}\,dt + \sqrt{\beta_{\min} + t(\beta_{\max} - \beta_{\min})}\,d\mathbf{w}, \quad t \in (0,1). \tag{4.24}$$

In conclusion, the contents above indicate that

For **SMLD (Variance Exploding SDE - VE)**:

- $f(x,t) = 0$
- $g(t) = \sigma_{\min}\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \sqrt{2\log\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)}$

For **DDPM (Variance Preserving SDE - VP)**:

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

4.6.2 Continuos Backward Process - PC Sampler

Here we can of course use the $f(x,t)$ and $g(t)$ to do the reverse process as equation (4.15) shows. However, here we possess additional insights that can enhance our solution methods. Specifically, with our score-based model $s_{\theta^*}(x,t) \approx \nabla_x \log p_t(x)$, we can utilize score-based Markov Chain Monte Carlo (MCMC) techniques, such as Langevin MCMC (Parisi, 1981; Grenander & Miller, 1994)to sample directly from the distribution p_t and refine the outputs of a numerical SDE solver.

At each time step, the numerical SDE solver provides an initial estimate for the sample at the next time step, functioning as a "predictor." Subsequently, the score-based MCMC method adjusts the estimated sample's marginal distribution, acting as

a "corrector." This approach is reminiscent of Predictor-Corrector methods, which are a class of numerical continuation techniques used for solving systems of equations (Allgower & Georg, 2012). We similarly refer to our hybrid sampling algorithms as Predictor-Corrector (PC) samplers.

The PC samplers extend the original sampling methodologies of SMLD and DDPM: the SMLD method employs an identity function as the predictor and utilizes annealed Langevin dynamics as the corrector. In contrast, the DDPM method adopts ancestral sampling as the predictor and the identity function as the corrector.

Algorithm 1 *PC sampling (VE SDE)*

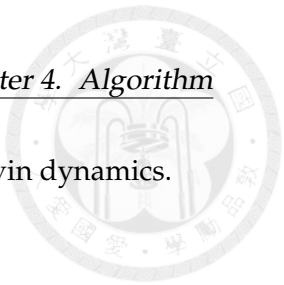
$\mathbf{x}_N \sim \mathcal{N}(0, \sigma_{\max}^2 \mathbf{I})$
for $i = N-1$ to 0 **do**
  $\mathbf{x}_i' = \mathbf{x}_i - g^2(t) s_\theta^*(\mathbf{x}_i, \sigma_i)\,\Delta t$
  $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$
  $\mathbf{x}_i = \mathbf{x}_i' + g(t)\sqrt{\Delta t}\mathbf{z}$
  for $j = 1$ to M **do**
    $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$
    $\mathbf{x}_i \leftarrow \mathbf{x}_i + \epsilon_i s_\theta^*(\mathbf{x}_i, \sigma_i) + \sqrt{2\epsilon_i}\mathbf{z}$
  end for
end for
return $\mathbf{x}_0$

Algorithm 2 *PC sampling (VP SDE)*

$\mathbf{x}_N \sim \mathcal{N}(0, \mathbf{I})$
for $i = N-1$ to 0 **do**
  $\mathbf{x}_i' = (f(x,t) - g^2(t) * s_\theta^*(\mathbf{x}_{i+1}, i+1))\Delta t$
  $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$
  $\mathbf{x}_i = \mathbf{x}_i' + g(t)\sqrt{\Delta t}\mathbf{z}$
  for $j = 1$ to M **do**
    $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$
    $\mathbf{x}_i \leftarrow \mathbf{x}_i + \epsilon_i s_\theta^*(\mathbf{x}_i, i) + \sqrt{2\epsilon_i}\mathbf{z}$
  end for
end for
return $\mathbf{x}_0$

where ϵ is defined as

$$\epsilon = 2r^2 \frac{\|\mathbf{z}\|_2^2}{\|s_\theta\|_2^2} \tag{4.25}$$



and r is a hyperparameter that controls the step size of the Langevin dynamics.

4.7 Conclusion

Chapter 5

Model Structure

In the previous chapter, we introduced the foundational algorithms employed in this research project. This chapter delves into the structure of our custom Transformer-based model, designed to predict the "score" or gradient in detector simulations. Built upon the Transformer architecture—a cutting-edge model in deep learning—our model incorporates several modifications to enhance its applicability in high-energy physics detector simulations.

We chose the Transformer architecture not only for its power and versatility but also for its unique suitability for data with rotational invariance. In our research, each input consists of multiple showers, each shower containing several hits, and each hit represented by four features, as introduced in Chapter 3. This structure makes our data rotationally invariant, meaning that the relationships within the data remain consistent even if the order of hits within a shower or the showers within an input are rearranged. Transformers are particularly well-suited for handling such properties. Their self-attention mechanism enables them to learn and capture relationships between data points in a way that is invariant to transformations like rotation. This flexibility is especially advantageous for our detector simulations, where capturing invariant relationships is crucial for making accurate predictions.

We will begin by exploring the evolution of Transformers from Recurrent Neural Networks (RNNs), highlighting how Transformer architectures overcame the limitations of sequential models. Following this, we will examine the core components of the Transformer model, including its different architectural types (encoder-only, decoder-only, and encoder-decoder models) and the self-attention mechanism, which lies at the heart of the Transformer's ability to model long-range dependencies.

After establishing an understanding of the original Transformer architecture, we

will discuss the custom modifications introduced in our model to optimize it for detector simulations. Key innovations include the **Gaussian Fourier Projection** for encoding temporal information, which allows the model to capture high-frequency dependencies by transforming time and incident energy into sinusoidal features. Additionally, we introduce a specialized **mean-field attention mechanism**, a variant of self-attention tailored to efficiently aggregate global context. Mean-field attention leverages a class token to summarize information across the sequence, reducing computational complexity while retaining essential global information.

Furthermore, our model incorporates residual network structures and layer normalization to stabilize and expedite the training process. We will explain how these modifications, along with our encoder-only architecture, facilitate efficient information flow, enabling the model to focus on capturing the relationships within the data rather than generating sequences. We also employ **Weights & Biases (wandb)** for parameter tuning, using its sweep functionality to systematically explore hyperparameters such as the number of encoder blocks, attention heads, and dropout rates to achieve optimal performance.

In summary, this chapter provides a comprehensive overview of our custom Transformer model, from its foundational components to the innovative adjustments that make it well-suited for high-energy physics applications. Through these design choices, our model efficiently captures both local and global dependencies, thereby enhancing the accuracy and fidelity of detector simulations.

5.1 Transformer

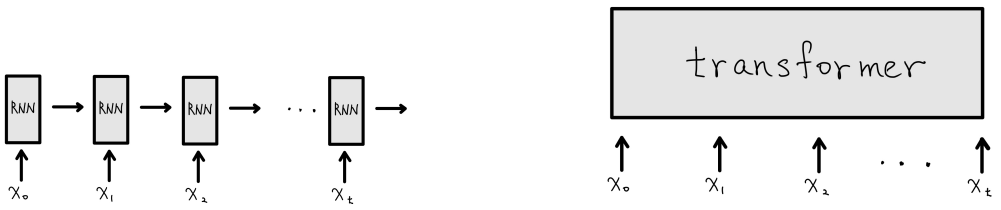
5.1.1 Introduction

Transformer models have transformed deep learning applications across various domains, providing significant advantages in handling complex and large datasets. In high-energy physics, where data from detectors is vast and multidimensional, advanced models like Transformers enhance both the accuracy and efficiency of simulations designed to emulate particle collisions and energy depositions within detectors.

5.1.2 The Evolution from RNNs to Transformers

Prior to Transformers, Recurrent Neural Networks (RNNs) were widely used for sequence modeling due to their ability to capture dependencies between sequential elements. However, RNNs are inherently sequential, making them slow and prone to issues like vanishing gradients, particularly with long sequences.

The introduction of Transformers by Vaswani et al. in their seminal paper, *"Attention is All You Need,"* addressed these limitations by introducing the self-attention mechanism. This innovation enables Transformers to capture long-range dependencies without the need for recurrent connections, allowing for faster and more efficient training.



(A) RNN Model (B) Transformer Model

FIGURE 5.1: Comparison of RNN and Transformer architectures.

5.1.3 Types and Structure of Transformer Architectures

The original Transformer architecture, as introduced by Vaswani et al., consists of both an encoder and a decoder. The encoder processes the input sequence, while the decoder generates the output sequence. This setup is particularly effective for tasks like machine translation. However, in practice, different applications benefit from using only the encoder or decoder.

The three main types of Transformer architectures are as follows:

- **Encoder-only Models**: Encoder-only models, such as BERT, create contextual embeddings by attending to all tokens bidirectionally. These models are ideal for tasks requiring sequence understanding, like classification.
- **Decoder-only Models**: Decoder-only models, like GPT, are designed for unidirectional generation. Each token attends only to previous tokens, making these models suitable for tasks like language modeling.
- **Encoder-Decoder Models**: The original Transformer model combines both an encoder and a decoder, making it effective for sequence-to-sequence tasks such as machine translation. Examples include BART and T5.

5.1.4 Choosing an Encoder-Only Model for Detector Simulation

In the context of detector simulation, our objective is to generate a high-quality representation of input data, such as particle collisions, without generating new sequences.

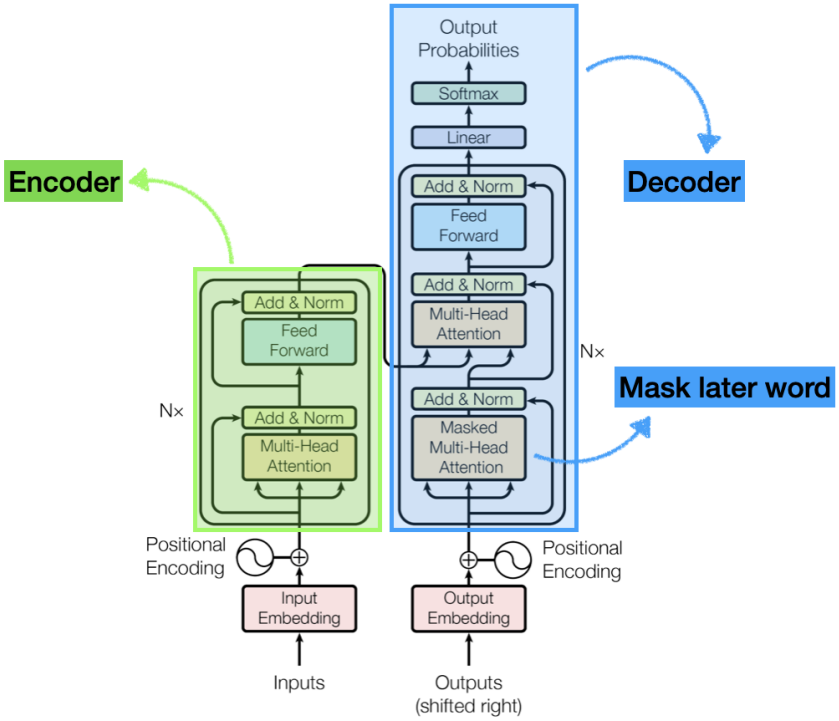


FIGURE 5.2: The structure of the original Transformer model. Adapted from *"Attention is All You Need,"* with additional annotations.

Thus, an encoder-only model is more appropriate, as it efficiently processes and encodes input data, capturing necessary features without the additional complexity of a generative decoder.

5.2 Self-Attention Mechanism

Self-attention is central to the Transformer model, enabling each token in a sequence to attend to all other tokens. For each token, the attention scores are computed based on the query Q, key K, and value V vectors:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.1}$$

This mechanism allows the model to capture relationships between distant tokens, which is crucial for high-dimensional data, such as particle detector simulations. Figure ?? illustrates the self-attention mechanism, showcasing how each token dynamically weights other tokens in the sequence.

5.3 Our Model Structure

Our model architecture builds upon the Transformer framework, with specific modifications to optimize performance in detector simulations, as shown in Figure ??.

We incorporate **Gaussian Fourier Projections** [22] to effectively encode temporal information, dense layers to transform conditional variables, and **mean-field attention** [23] to efficiently aggregate global context. These architectural choices enable our model to capture complex dependencies, thereby enhancing the fidelity and accuracy of simulation outcomes.

5.3.1 Gaussian Fourier Projection for Temporal Encoding

The Gaussian Fourier Projection component encodes temporal information using Gaussian random features. This technique allows the model to incorporate high-frequency time-dependent information, in our case time and incident energy, which is crucial for capturing the dynamics of particle interactions within detectors.

In our model, we apply a Fourier feature mapping γ to featurize input coordinates before passing them through a coordinate-based multilayer perceptron (MLP). This approach improves both convergence speed and generalization.

The mapping function γ transforms input points $\mathbf{v} \in [0,1)^d$ onto the surface of a higher-dimensional hypersphere using sinusoidal functions:

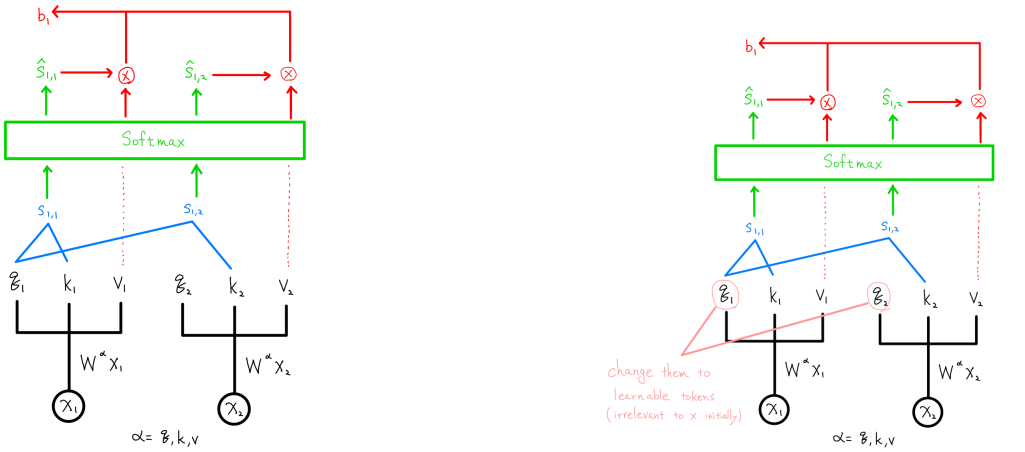
$$\gamma(\mathbf{v}) = \begin{bmatrix} a_1 \cos(2\pi \mathbf{b}_1^T \mathbf{v}) \\ a_1 \sin(2\pi \mathbf{b}_1^T \mathbf{v}) \\ \vdots \\ a_m \cos(2\pi \mathbf{b}_m^T \mathbf{v}) \\ a_m \sin(2\pi \mathbf{b}_m^T \mathbf{v}) \end{bmatrix} \tag{5.2}$$

where a_i and $\mathbf{b}_i$ are parameters that control the scaling and frequency of each sinusoid. We set $a = 1$ for all cases and experiment with different values of $\mathbf{b}$ to identify optimal performance. The results are presented in subsequent sections.

5.3.2 Mean-Field Attention in Detector Simulation

Our model utilizes a variation of self-attention called **mean-field attention**. Unlike traditional self-attention, mean-field attention employs a class token to aggregate information from all tokens, creating a global summary. This reduces computational complexity while preserving essential global context.

Mean-field attention allows the class token to encapsulate the sequence's essential features by attending to each token once. This mechanism is computationally efficient and well-suited for high-energy physics applications, where capturing global properties of particle collisions is more important than individual token interactions. Figure ?? provides a comparison between self-attention and mean-field attention mechanisms.



(A) Self-Attention Mechanism (B) Mean-Field Attention Mechanism

FIGURE 5.3: Comparison of self-attention and mean-field attention mechanisms.

5.3.3 Parameter Tuning

Once the model structure is established, tuning the parameters is essential for optimal performance.

To optimize the model, we utilize **Weights & Biases (wandb)** for parameter tuning. Using wandb's sweep functionality, we systematically explore hyperparameters, including:

- **Number of blocks**: Controls the depth of the Transformer model.
- **Number of heads**: Determines the number of attention heads in the multi-head attention mechanism.
- **Hidden dimension**: Sets the size of the hidden layers in the model.
- **Embed dimension**: Specifies the embedding size for the model's input.
- **Batch size**: Number of samples per batch.
- **Learning rate**: Determines the rate at which the model updates during training.

- **Dropout rate**: The fraction of nodes dropped during training to prevent overfitting.
- **Sampling steps**: The number of sampling steps when solving SDE.
- **Correction steps**: The number of correction steps in each sampling step.
- **Scale in Fourier features**: The scaling factor for Fourier features.

The results of this tuning process are presented in later chapters.

5.4 Conclusion

Our custom Transformer model leverages specialized architectural choices to optimize performance in high-energy physics simulations. Key modifications include **Gaussian Fourier projections** for encoding time and incident energy, and **mean-field attention** for capturing global context beyond immediate shower information. The addition of a class token enables the model to represent both local and global dependencies, making it particularly suitable for scenarios with strong temporal and energetic relationships.

The mean-field attention mechanism enhances computational efficiency by reducing complexity while preserving essential global information. Parameter tuning plays a crucial role in achieving optimal performance, as we demonstrate with our use of wandb. By employing an encoder-only model, we capture inter-token relationships within the data, making our approach well-suited for high-energy physics applications.

Chapter 6

Strategies and Results

6.1 Metrics

6.1.1 FID Score

To evaluate the performance of our model, we employed the Fréchet Inception Distance (FID) score as a key metric. The FID score is widely used to assess the quality of generated samples by measuring the distance between the feature representations of real and generated images using the InceptionV3 model [**inceptionv3**]. A lower FID score indicates that the generated samples are closer to the real samples in terms of their statistical distribution. We utilized the PyTorch library's implementation of the FID score [**pytorch**] for our calculations.

6.1.2 Classifier

6.2 VE and VP Studies

To begin, we compared the performance of the Variance Exploding (VE) and Variance Preserving (VP) methods. Both methods were used to train the model, and their results were evaluated. First, we observed that the loss curves during training exhibit distinct shapes for the two methods.

FIGURE 6.1: Comparison of training loss curves for VE and VP methods.

We also compared the FID scores of models trained with the VE and VP methods. The results showed that the VE method resulted in a lower FID score compared to the VP method. This suggests that the VE method is more effective at pushing the model toward generating random samples that better represent the initial sampling space.

FIGURE 6.2: Comparison of FID scores for VE and VP methods.

In conclusion, the VE method outperformed the VP method in terms of FID score. We guess this is because it has more power to push our data to random noise, which is

the initial state of sampling space. So our model know how to do the reverse process at the beginning in VE method. For example, if we see the standard deviation of both VE and VP methods, one can find out VE has the steeper slope than VP, which means it has the power to push the data to the random noise.

FIGURE 6.3: Comparison of FID scores for VE and VP methods.

6.3 Parameter Sweeping

To optimize the model, we conducted a parameter sweeping study using `wandb`. We experimented with various learning rates, batch sizes, and hidden dimensions. Our findings indicated that the best-performing parameter configuration was:

- Learning rate: 0.0003
- Batch size: 128
- Embedding dimension: 96
- Hidden dimension: 96

FIGURE 6.4: Visualization of parameter sweeping results.

6.4 Energy Distribution

With the optimal settings, our model was able to generate the basic shapes of both the energy and spatial distributions. However, the model often produced an excessive number of hits (`nhits`) at higher energy levels, leading to overestimation. This issue was not observed when training on single-bucket data, indicating that the model struggles to differentiate between data from different buckets. This suggests that our conditional variables are not functioning effectively.

6.5 Centralization

In addition to the conditional variable issue, visualizing 2D or 3D plots revealed that the model failed to capture the relationship between energy and radius. A key observation was that the model could not learn that higher energy values should be concentrated near the center (smaller radii). Consequently, while the 1D plots were satisfactory, the generated samples lacked proper centralization.

To address this, we first tried to transform our data into spherical coordinate and introduce a correlation term between energy and theta in the loss function to try to

suppress relation between energy and theta, hoping our model can thus learn more about the relation between energy and radius.

The new loss function is defined as:

$$L = L_{\text{MSE}} + \lambda L_{\text{cor}}, \tag{6.1}$$

where L_{MSE} is the mean squared error loss, L_{cor} is the correlation loss, and λ is a weighting factor for the correlation loss. The correlation loss is defined as:

$$L_{\text{cor}} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}), \tag{6.2}$$

where x and y are the variables of interest, and $\bar{x}$ and $\bar{y}$ are their respective means.

The reason why we don't apply the correlation term between energy and radius is that the relation between them is by experienced, although everyone would expect the result, it's not solid, we don't want to bias our model, or you can say we don't want to tell the answer of the relation to our model.

However, although the correlation term was added to the loss function and it indeed suppressed the relation between energy and theta, the centralization of the generated samples did not improve significantly. This suggests that the correlation term alone is not sufficient to address the centralization issue.

FIGURE 6.5: The Picture after adding the correlation term.

Chapter 7

Future Goals

Looking ahead, there are two primary objectives for future work:

7.1 Further Acceleration of the Model

The first goal is to further improve the speed of the model. Currently, our model achieves a 100x speedup compared to Geant4 simulations. However, there is potential for even greater acceleration by exploring alternative methods. For instance, replacing the Stochastic Differential Equation (SDE) framework with an Ordinary Differential Equation (ODE) approach, or implementing a restart method as suggested in [**restart**], could lead to significant improvements in computational efficiency.

7.2 Layer Relationship Learning and Tracking

The second goal is to enhance the model's ability to learn the relationships between layers. Specifically, we aim to train the model to identify which hits in one layer correspond to hits in the previous layer. This capability would enable the development of a model for particle tracking, providing a more comprehensive and detailed understanding of the underlying physical processes.

Achieving these goals would not only improve the current model but also open new possibilities for its application in simulation and analysis.

Bibliography

[1] Lyndon Evans and Philip Bryant. "LHC Machine". In: *Journal of Instrumentation* 3.08 (2008), S08001. DOI: 10.1088/1748-0221/3/08/S08001.

[2] The Atlas Collaboration et al. "The ATLAS Experiment at the CERN Large Hadron Collider". en. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08003–S08003. ISSN: 1748-0221. DOI: 10.1088/1748-0221/3/08/S08003.

[3] The CMS Collaboration et al. "The CMS experiment at the CERN LHC". In: *Journal of Instrumentation* 3.08 (2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004.

[4] G Apollinari et al. *High Luminosity Large Hadron Collider HL-LHC*. en. 2015. DOI: 10.5170/CERN-2015-005.1.

[5] S. Agostinelli et al. "Geant4—a simulation toolkit". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506.3 (2003), pp. 250–303. ISSN: 0168-9002. DOI: https://doi.org/10.1016/S0168-9002(03)01368-8.

[6] Y. Song and S. Ermon. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *arXiv preprint arXiv:2011.13456* (2020).

[7] V. Mikuni and B. Nachman. "CaloScore: A Conditional Generative Model for Calorimeter Shower Simulation". In: *arXiv preprint arXiv:2106.00792* (2021).

[8] S. Agostinelli et al. "Geant4: A Simulation Toolkit". In: *Nuclear Instruments and Methods in Physics Research Section A* 506.3 (2003), pp. 250–303.

[9] J. Allison et al. "Geant4 Developments and Applications". In: *IEEE Transactions on Nuclear Science* 53.1 (2006), pp. 270–278.

[10] J. Allison et al. "Recent Developments in Geant4". In: *Nuclear Instruments and Methods in Physics Research Section A* 835 (2016), pp. 186–225.

[11] I. Goodfellow et al. "Generative Adversarial Networks". In: *arXiv preprint arXiv:1406.2661* (2014).

[12] D. P. Kingma and M. Welling. "Auto-Encoding Variational Bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[13] L. Dinh, J. Sohl-Dickstein, and S. Bengio. "Density Estimation Using Real NVP". In: *arXiv preprint arXiv:1605.08803* (2016).

[14] M. Paganini, L. de Oliveira, and B. Nachman. "CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks". In: *Physical Review D* 97.1 (2018), p. 014021. DOI: 10.1103/PhysRevD.97.014021.

[15] ATLAS Collaboration. *Fast Calorimeter Simulation with Generative Adversarial Networks*. Tech. rep. ATL-SOFT-PUB-2018-001, 2018.

[16] S. Verheyen and B. Krislock. "CaloFlow: Fast and Accurate Generation of Calorimeter Showers with Normalizing Flows". In: *arXiv preprint arXiv:2106.05285* (2021).

[17] S. Verheyen and B. Krislock. "CaloFlow II: Even Faster and Still Accurate Generation of Calorimeter Showers with Normalizing Flows". In: *arXiv preprint arXiv:2107.13684* (2021).

[18] CMS Collaboration. *The CMS High Granularity Calorimeter for HL-LHC Upgrade*. Tech. rep. CERN-LHCC-2017-023, 2017.

[19] CMS Collaboration. "Design and Performance of the CMS Beam Radiation, Instrumentation, and Luminosity Detectors". In: *Journal of Instrumentation* 13.10 (2018), P10034.

[20] Yang Song and Stefano Ermon. *Generative Modeling by Estimating Gradients of the Data Distribution*. arXiv:1907.05600. Oct. 2020.

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851.

[22] Matthew Tancik et al. *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*. en. arXiv:2006.10739 [cs]. June 2020.

[23] Benno Käch, Isabell Melzer-Pellmann, and Dirk Krücker. *Pay Attention To Mean Fields For Point Cloud Generation*. en. arXiv:2408.04997 [hep-ex]. Aug. 2024.