# Score-based Generative Models for Detector Reconstruction and Fast Simulations in High-Energy Physics

Undergraduate Thesis

*Submitted in partial fulfillment of the requirements for*

**BITS 421T/422T Thesis**

*by*

**Ameya Thete**
**2018B5A70885G**

Under the supervision of

**Dr. Sergei Gleyzer**
The University of Alabama, Tuscaloosa & Fermilab


**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI - KK BIRLA GOA CAMPUS**

# Declaration of Authorship

I, Ameya Thete, declare that this undergraduate thesis titled *Score-based Generative Models for Detector Reconstruction and Fast Simulations in High-Energy Physics*, and the work presented in it are my own. I also certify that:

- This work was wholly completed while in candidature for a research degree at BITS, Pilani - KK Birla Goa Campus;
- This thesis has previously not been submitted to any degree or other qualification at BITS, Pilani - KK Birla Goa Campus or any other institution;
- When I have consulted the published work of others, this is always clearly attributed.
- When I have quoted from the work of others, the source is always attributed. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

**Date:** 15$^{th}$ May, 2023

# CERTIFICATE

This is to certify that the thesis titled *Score-based Generative Models for Detector Reconstruction and Fast Simulations in High-Energy Physics*, submitted by Ameya Thete (2018B5A70885G) in partial fulfillment of the requirements of BITS 421T/422T Thesis embodies the work done by him under my supervision.

---

*Supervisor*
Dr. Sergei Gleyzer
The University of Alabama, Tuscaloosa &
Fermilab

**Date:** 15th May, 2023

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI - KK BIRLA GOA CAMPUS

# *Abstract*

Master of Science (Physics) & Bachelor of Engineering (Computer Science)

**Score-based Generative Models for Detector Reconstruction and Fast Simulations in High-Energy Physics**

by Ameya Thete

In recent years there has been considerable progress in developing machine learning models suitable for applications in high-energy physics (HEP) for tasks such as event simulation, jet classification, and anomaly detection. In particular, there is a pressing need to develop faster and more accurate techniques for simulating particle physics processes. Currently, such simulations are both time-intensive and require heavy computational resources. Moreover, the High-Luminosity LHC (HL-LHC) upgrades are expected to place the existing computational infrastructure under unprecedented strain due to increased event rates and pileups. Simulations of particle physics events need to be faster without negatively affecting the accuracy and fidelity of the results. Recently, score-based generative models have been shown to produce realistic samples even in large dimensions, surpassing current state-of-the-art models on different benchmarks and categories. To this end, we introduce a score-based generative model in collider physics based on thermodynamic diffusion principles that provides effective reconstruction of LHC events on the level of calorimeter deposits and tracks, which offers the potential for a full detector-level fast simulation of physics events. We work with denoising diffusion probabilistic models (DDPMs) and adapt them to a point cloud based representation of low-level detector data to faithfully model the distribution of hits in the barrel region of the electromagnetic calorimeter (ECAL) of the Compact Muon Solenoid (CMS) detector array. While this work is limited to its applications for the CMS detector suite, the point cloud formulation allows the method to readily be extended to alternative detector geometries.

**Keywords**: High-Energy Physics, Deep Learning, Diffusion, Point Clouds

# ACKNOWLEDGEMENTS

I would like to express my profound gratitude to have had the opportunity to work on such an interesting and exciting topic for my B.E.-M.Sc thesis with the CMS experiment at CERN. I deeply thank my advisor Dr Sergei Gleyzer for his unwavering support and supervision throughout the project. Our discussions and meetings have always been encouraging and have always renewed my excitement and passion towards the project. I will always remember his mentoring as I advance to new stages in my career. I'm also grateful to my colleagues in the end-to-end (E2E) project, particulary Tom Magorsch for his help and insightful comments that were crucial to my work. Back home, I'm thankful to Dr Raghunath Ratabole and Dr Radhika Vathsan for instilling an interest and appreciation for high-energy physics and their support as I navigated through my undergraduate degree. I am grateful the Department of Science and Technology, Government of India for financial support during my undergrad through the INSPIRE SHE program. Finally, I thank the Anuradha and Prashanth Palakurthi Centre for Artificial Intelligence Research (APPCAIR) at BITS Goa for providing access to their high-performance computing environment that heavily contributed to the results in this work.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The goal of particle accelerators, like the Large Hadron Collider (LHC), is to perform extensive studies of the subatomic nature of matter by searching for new particle states, validating and probing the limits of existing theories, and testing newer ones. The LHC, in particular, works by accelerating beams of protons in opposite directions and colliding them at four major interaction points: CMS, ATLAS, LHCb, and ALICE. After a collision, most particles produced in the event readily decay into relatively stable species which leave characteristic hits within a wide array of detectors. We can then analyze the particle showers that emerge from these hits to perform an in-depth analysis to extract relevant information associated with the theory being probed. The sheer complexity introduced by many hardware and software components working in tandem makes the LHC a highly expensive (both computationally and fiscally) endeavour to operate. This introduces the need to perform extensive simulations of events that we expect to see, *before* actually running a particle collision. Several simulation software suites have been developed for HEP uses. GEANT4 [1] is one such simulation toolkit that relies on full simulation techniques accounting for particle-detector interactions and detector geometries.

Despite their unquestionably pivotal role in modern HEP analyses, these simulation techniques require extensive computational resources and are often detector-specific. The advent of the High Luminosity LHC (HL-LHC) is expected to place a heavy strain on our current resources, which compels the need to look for faster and more resource-efficient simulation tools and algorithms. Deep generative models have already experienced burgeoning popularity and use in many scientific and non-scientific domains for a variety of use cases, and as such are a promising candidate for HEP applications. Many works [2, 3, 4, 5, 6] have already demonstrated how deep generative models can be used for HEP use cases or integrated into a typical HEP analysis pipeline [7]. This thesis endeavours to take a step in the same direction.

In this thesis work, we propose to use a (slightly generalized form of a) class of iterative deep generation models called denoising diffusion probabilistic models (DDPMs) to learn the representations of jet events resulting from proton-proton collisions. Instead of simulating jets in

terms of more physical variables like jet mass or momentum, we instead work with low-level representations and simulate the response of detectors to these jets. A point cloud jet representation allows for a more efficient treatment of data, and also abstracts away the geometry of the detector in question. This work only limits itself to the discussion of the potential of state-of-the-art deep learning techniques for use in HEP, and does not go into specific details of how one might go about integrating such models within the existing analysis pipeline. This thesis is structured as follows:

In Chapter 2, we begin by discussing, albeit briefly, the theory behind modern particle physics. We elucidate the structure of perhaps the most successful theory in physics, the Standard Model of Particle Physics, by providing a broad overview of its various sectors. We then conclude by discussing the various extensions proposed to the Standard Model that attempt to explain new physics beyond the Standard Model.

In Chapter 3, we introduce the experimental apparatus, i.e., the Large Hadron Collider, which is responsible for collecting the data that is used for modern particle physics calculations. We also introduce the CMS experimental setup, and describe its detector apparatus, whose response we are trying to simulate.

In Chapter 4, we introduce the mathematics of how deep learning works, gradient descent (which is the optimization algorithm critical to most modern deep learning architectures), followed by a coverage of improvements to gradient descent that are popularly used today. We conclude this chapter by introducing generative deep learning models, how they are optimized, and the variational autoencoder, which is the simplest variational generative model.

In Chapter 5, we propose score-based diffusion models as an alternative methodology for high-energy physics simulations. This chapter delves deep into the mathematical formulation of a class of score-based models called denoising diffusion probabilistic models, and then goes on to generalize them to a continuous time domain. We conclude by showing how these models are trained in practice, and used to generate samples from pure noise.

In Chapters 6 and 7, we introduce the dataset used to train our diffusion models, and specific architectural details about the diffusion framework. The majority of this chapter is devoted to describing the results we obtain, and evaluating our results across a range of metrics. Finally, we conclude by indicating possible research directions that can be undertaken in the future to improve upon the results obtained in this thesis.

# Chapter 2

# Theoretical Foundations

> "If I could remember the names of these particles, I would have been a botanist."
>
> Enrico Fermi

The field of high energy physics (HEP) attempts to answer some of the most basic (and also rather existential) questions underlying our understanding of the universe: What is matter? How can matter and its interactions explain phenomena we observe all around us? Although the study of matter itself is not new — theories postulating the composition and origin of matter can be traced all the way back to the $6^{\text{th}}$ century BCE — technological advances over the past century have allowed us to study physics at unprecedented length and energy scales. The discovery of the sheer number of particles discovered in the mid-$20^{\text{th}}$ century led to the formalization of high-energy physics in the context of quantum field theory and marked the beginning of modern particle physics.

In this chapter, we will introduce the Standard Model of particle physics (SM), which describes all presently known fundamental particles and their interactions. Although an understanding of the mathematical description of the SM is not required to appreciate the results presented later in this thesis, it provides a broader context to our discussion. Moreover, we would be remiss to not include a discussion of one of the most successful theories in modern physics.

## 2.1 The Standard Model of Particle Physics

As indicated earlier, the Standard Model is perhaps one of, if not, the most successful theories of modern physics. It is incredibly robust to experimental results that have tried to probe at

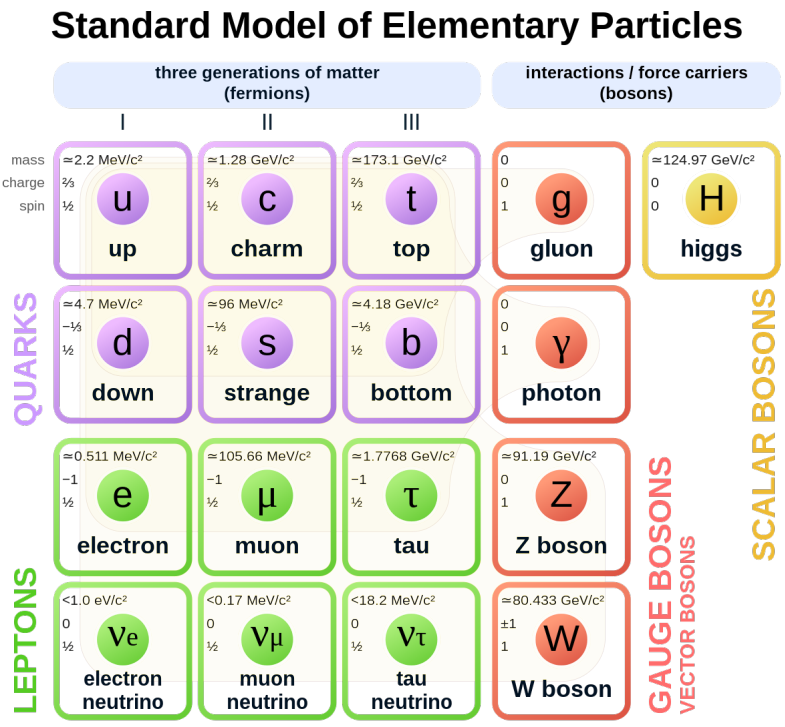


Figure 2.1: The Standard Model of particle physics.

its structure: countless results in particle physics from the past decade or so have shown no signs of deviation from SM predictions [1].

The Standard Model (SM) of particle physics is a collection of well-tested quantum field theories [8] that describe fundamental particles and their interactions with each other. More specifically, the SM provides a Lagrangian density for each sector, or sub-unit of the SM. The Lagrangian density describing any set of particles can be used to derive their equations of motion via the principle of least action. Particles in the SM are described as fundamental excitations of fields that pervade four-dimensional spacetime, and interactions between particles follow from restrictions imposed by gauge invariance on these fields. The SM describes three of the four fundamental forces of nature: the electromagnetic interaction, the strong interaction, and the weak interaction – all within the framework of quantum field theory. Although the SM does not describe the gravitational force, on the mass and length scales involved in most high-energy processes, the influence of gravity is negligible can often be ignored.

According to the spin-statistics theorem of quantum mechanics, fundamental particles can be categorized into two classes based on the value of their intrinsic spin: bosons and fermions. Bosons are particles that obey Bose-Einstein statistics [9] and have integral values of intrinsic spin, while fermions are particles that obey Fermi-Dirac statistics [10] and carry half-integral values of intrinsic spin. Depending upon the types of interactions they participate in, fermions

[1] So much so that the phrases "[...] is consistent with Standard Model predictions." or "no excesses were found" are infamously familiar to most particle physicists.

can further be divided into quarks and leptons, which in turn, are arranged into three generations according to their masses. Quarks undergo interactions via all fundamental forces, whereas leptons do not undergo interactions via the strong force. Basic properties of fermions are listed in Table 2.1. All bosons in the SM (with the exception of the Higgs) have spin 1 and are called gauge bosons. In the SM, forces are mediated by gauge bosons; that is, when two particles feel a force, it is common to say that they do so by exchanging a gauge boson corresponding to the relevant force. This particle-exchange picture of how forces work, though highly unintuitive, arises from the underlying mathematics of how forces are described by the SM. An overview of the basic properties of gauge bosons are listed in Table 2.2. Figure 2.1 illustrates the elementary particles described by the SM, their basic properties, and their classification.

| Fermion | Electric Charge (in $e$) | $I_Z$ | Mass |
|---|---|---|---|
| electron neutrino $\nu_e$ | 0 | $+1/2$ | $< 1.10$ eV |
| electron $e$ | $-1$ | $-1/2$ | 0.51 MeV |
| muon neutrino $\nu_\mu$ | 0 | $+1/2$ | $< 1.10$ eV |
| muon $\mu$ | $-1$ | $-1/2$ | 105.66 MeV |
| tau neutrino $\nu_\tau$ | 0 | $+1/2$ | $< 1.10$ eV |
| tau $\tau$ | $-1$ | $-1/2$ | $1776.86 \pm 0.12$ MeV |
| up quark $u$ | $+2/3$ | $+1/2$ | $2.20^{+0.50}_{-0.40}$ MeV |
| down quark $d$ | $-1/3$ | $-1/2$ | $4.70^{+0.50}_{-0.30}$ MeV |
| charm quark $c$ | $+2/3$ | $+1/2$ | $1.28^{+0.03}_{-0.04}$ GeV |
| strange quark $s$ | $-1/3$ | $-1/2$ | $95^{+9}_{-3}$ MeV |
| top quark $t$ | $+2/3$ | $+1/2$ | $173^{+0.40}_{-0.40}$ GeV |
| bottom quark $b$ | $-1/3$ | $-1/2$ | $4.18^{+0.04}_{-0.03}$ GeV |

Table 2.1: The electric charge, the third component of the weak isospin $I_Z$, and the mass of the elementary fermions of the SM. The quark and lepton generations are separated by horizontal lines. These values are taken from [11].

| Boson | Interaction | Electric Charge (in $e$) | $I_Z$ | Mass |
|---|---|---|---|---|
| Photon $\gamma$ | electroweak | 0 | 0 | massless |
| $W^\pm$ boson | electroweak | $\pm 1$ | $\pm 1$ | $90.38 \pm 0.01$ GeV |
| $Z^0$ boson | electroweak | 0 | 0 | $91.19 \pm 0.00$ GeV |
| 8 gluons $g$ | electroweak | 0 | 0 | massless |

Table 2.2: The interaction, the electric charge, the third component of the weak isospin $I_Z$, and the mass of the gauge bosons of the SM are listed. These values are taken from [11]

### 2.1.1 The Electroweak Interaction

Although the electromagnetic and weak interactions are sometimes listed separately, the electroweak (EW) theory [12, 13, 14] proposed by Sheldon Glashow, Abdus Salam, and Steven Weinberg showed that the two forces can be described by a unified interaction called the electroweak interaction. The electromagnetic component of the EW theory – called Quantum

Electrodynamics (QED) – is an Abelian gauge theory [2] with a U(1) symmetry group and a photon as the gauge boson for the theory [15]. The coupling strength $\alpha$ of the photon to fermions is proportional to the electric charge $q$ of the interacting entities and thus affects only charged fermions. The weak component of the EW theory is described by a $\mathrm{SU(2)_L}$ symmetry group which couples to the third component of isospin, $I_Z$. All fermions are subjected to the weak interaction, and the force is mediated by the $\mathrm{W}^{\pm}$ bosons and the $\mathrm{Z}^0$ boson. $W$ bosons allow quarks to change flavours, while the $Z$ boson allows changes in spin, momentum and energy only.

The electroweak interaction is therefore described by an $\mathrm{SU(2)_L} \times \mathrm{U(1)}$ symmetry group resulting in four massless gauge bosons: $\mathrm{B}^0$, $\mathrm{W}^0$, $\mathrm{W}_1$, and $\mathrm{W}_2$. The four physical gauge bosons $\gamma$, $\mathrm{W}^{\pm}$, and $\mathrm{Z}^0$ are related to these gauge bosons as

$$\begin{pmatrix} \gamma \\ Z \end{pmatrix} = \begin{pmatrix} \cos\theta_W & \sin\theta_W \\ -\sin\theta_W & \cos\theta_W \end{pmatrix} \begin{pmatrix} \mathrm{B}^0 \\ \mathrm{W}^0 \end{pmatrix} \tag{2.1}$$

$$\mathrm{W}^{\pm} = \frac{1}{\sqrt{2}}(W_1 \mp iW_2) \tag{2.2}$$

where $\theta_W$ is called the weak mixing angle or the Weinberg angle.

### 2.1.2 Quantum Chromodynamics

The strong interaction is described in the SM by the quantum field theory called Quantum Chromodynamics (QCD). QCD is a non-Abelian theory with a SU(3) symmetry group [16, 17]. The strong coupling $\alpha_s$ is proportional to the colour charge – the corresponding analogue to the electric charge in QED. There are three different colour charges: red, blue, and green (along with their respective anticolours). The theory is mediated by eight self-interacting, massless bosons called gluons. The gluons themselves carry colour charge. Using a semi-classical approach, we can estimate the strong potential to approximately

$$V(r) \propto \frac{4}{3}\frac{\alpha_s(r)}{r} + k \cdot r \tag{2.3}$$

where $r$ is the distance between two colour-charged particles and $k$ is a constant. This relation leads to an increasing potential energy as $r$ increases. If the distance between two particles exceeds a threshold, the potential energy grows large enough to generate a new quark-antiquark pair. As a result, there are no free colour-charged particles. This is called confinement. For small distances $r$ corresponding to large momentum transfers, the potential energy becomes small enough and leads to a weak interaction between the particles. This is called asymptotic freedom and can be used to describe QCD in a perturbative fashion.

[2] A group $(G, *)$ is said to be Abelian if the group operation $*$ is commutative, i.e, $\forall a, b \in G,\ a * b = b * a$

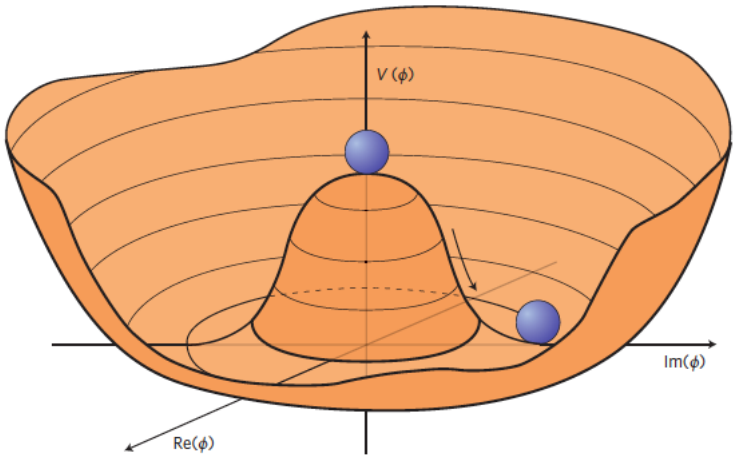


Figure 2.2: An illustration of the shape of the Higgs potential, $V(\phi)$, as predicted by the Standard Model. The ball indicates that the Higgs will settle into one of the infinitely many possible ground states in the trough of the potential with a non-zero expectation value [18].

### 2.1.3 The Higgs Sector

We conclude our discussion of the SM by describing its Higgs sector. It turns out that the equations of the SM are mathematically inconsistent unless there is an additional field called the Higgs field, with an associated particle called the Higgs boson. The Higgs is the only SM boson with no spin (and is thus called a scalar boson, and the corresponding field is called a scalar field). The Higgs field has a special property with interesting ramifications: its expectation value can be non-zero in vacuum. This is a consequence of the shape of the potential energy of the Higgs field [3], as shown in Figure 2.2

The SM predicts the Higgs potential to take the form

$$V(\phi) = a|\phi|^2 + b|\phi|^4 \tag{2.4}$$

with $a < 0$ and $b > 0$. In vacuum, through a process called "spontaneous symmetry breaking", the Higgs settles to one of the infinitely many possible ground states in the valley of this potential, where the value of the field is non-zero. The implications of the field having a non-zero value is that it gives fundamental particles their mass.

This, therefore, disproves the prediction that the $W^{\pm}$, and $Z^0$ bosons are massless. Higgs, Brout, and Englert proposed a solution to this problem in 1964 via the Higgs mechanism [19, 20, 21] which generates massive particles due to spontaneous symmetry breaking. The Higgs mechanism introduces a doublet of a complex scalar field to the SM which results in four additional degrees of freedom. Three of these degrees of freedom result in massless particles in accordance with the Goldstone theorem [22]. Due to spontaneous symmetry breaking,

[3] The potential energy can be thought of as the amount of energy required by the Higgs field to take on a certain value.

they are absorbed by the $W^{\pm}$, and $Z^0$ bosons, generating their masses

$$m_W^2 = \frac{v^2 g^2}{4} \tag{2.5}$$

$$m_Z^2 = \frac{1}{4}\left(g^2 + (g')^2\right) v^2 \tag{2.6}$$

while leaving the photon massless. $g$ and $g'$ denote the coupling constants of the weak charge and hypercharge, respectively. $v$ ($\approx 246\,\text{GeV}$) is the vacuum expectation value of the Higgs field. The coupling constants are related to the electric charge $e$ via the Weinberg angle $\theta_W$ as

$$e = g' \cos\theta_W = g \sin\theta_W \tag{2.7}$$

Fermionic masses $m_f$ are also generated by the Higgs mechanism via a Yukawa-type interaction [23]. This remaining degree of freedom is the Higgs boson (H). A particle compatible with the SM Higgs was discovered at the LHC by the CMS and ATLAS experiments in 2012 [8], largely believed to be so because the measured mass was consistent with the SM prediction for the Higgs mass $m_{\text{H}} \approx 125$ GeV.

## 2.2 Beyond The Standard Model

For all its strengths and robustness to experimental evidence, we have strong reason to believe that the Standard Model is not the complete picture of the universe, but is in fact a small subset of a more generalized theory. Moreover, observations collected from experiments at the LHC as well as certain theoretical considerations strongly suggest the existence of new particles at multi-TeV scales. In this section, we briefly cover some of the shortcomings of the Standard Model, as well as some proposed solutions to these problems. These issues and their solutions fall under the broad umbrella of Beyond the Standard Model Physics (BSM).

### 2.2.1 The Gravitational Force

Perhaps the most obvious shortcoming of the Standard Model is its elision of the gravitational interaction. At the time of writing this thesis, gravity is well-understood to be described by Albert Einstein's General Relativity, but this theory appears to be uncomfortable with much of the particle talk that surrounds the SM. While there is stark experimental and theoretical evidence that the SM is incomplete by itself, and that it somehow has an interplay with other theories, we have still been unable to reconcile the SM with gravitation. This active area of research falls under the domain of quantum gravity, which attempts to describe gravity according to the principles of quantum mechanics.

### 2.2.2 The Hierarchy Problem

The introduction of the Higgs mechanism to the SM tied up many loose ends, but it also introduced new complexities to the already precarious framework. The Higgs boson faces a unique problem where its measured mass is sensitive to physics at larger energy scales. Quantum field theories use a neat trick called renormalization, through which we can deal with infinities that arise during calculations by subtracting them away. One consequence of renormalization is that this links all parameters of a quantum field theory to some energy or distance scale; in other words, there is a threshold value of energy, beyond which the theory no longer applies. We can call that threshold $\Lambda_{\mathrm{UV}}$, and it turns out that the choice of this cutoff affects the value of the Higgs mass. This is because the Higgs boson gains its own mass in part due to the quantum fluctuations of many different fields, with $\Lambda_{\mathrm{UV}}$ controlling the scale of these fluctuations. As it appears, the Higgs mass is actually made up of two terms

$$m_H^2 \approx \left|a + \mathcal{O}(\Lambda_{\mathrm{UV}}^2)\right| \approx 125\,\mathrm{GeV} \tag{2.8}$$

where $a$ is a part of Eq. (2.4). This parameter, which is called the bare mass, has to be a large and negative value if $\Lambda_{\mathrm{UV}}$ is of any reasonable size in order to cancel out their values enough to get 125 GeV.

We can then say that the the Standard Model is no longer valid just above the current energy scale of the LHC (13.6 TeV), which would make the parameter $a$ be of a similar size such that their sum is 125 GeV. Then, we would have to see some newer theory to describe physics at the TeV scale, but we have not found any evidence for it so far. To completely rule out the possibility of any such new physics, we would have to assume it appears at scales much larger than this, which would lead to even larger numbers having to cancel out precisely to 125 GeV. While nature could just be fine-tuned, this open question is still a source of consternation for many physicists.

### 2.2.3 Supersymmetry

A commonly proposed theory to supersede the SM these days is supersymmetry, which makes it such that every particle in the SM has a supersymmetric partner at some higher energy scale. These additions to the SM help stabilize the mass of the Higgs boson the hierarchy problem described above, and also conveniently helps with the unification of forces. If you're wondering why we haven't rushed to adopt this perfect theory yet, it's due to a complete lack of experimental evidence. A major task of the CMS and ATLAS experiments at the LHC is to look for signatures of supersymmetry in the wild, to see if these hypothesized particles poke their heads out at any of our experiments.

# Chapter 3

# Experimental Setup: Particles and their Colliders

> "Data! Data! Data!", he cried impatiently. "I can't make bricks without clay."
>
> Sherlock Holmes, *The Adventure of the Copper Beeches*

Einstein's energy-mass equivalence implies that given sufficient energy, one might just be able to create matter with mass. Particle colliders work with the exact same idea: if we give particles enough energy and let them interact (via a controlled collision), we can then create newer, heavier particles! [1] The energy at which collisions occur is called the center-of-mass energy, which is around 13.6 TeV for Run 3 of the Large Hadron Collider.

In this chapter, we present an overview of the experimental setup used for most particle physics experiments. We first describe the setup of the Large Hadron Collider (LHC) followed by an overview of the Compact Muon Solenoid (CMS) experiment. This description would allow the reader to better understand the dataset that is used to train our models. We also briefly introduce Monte-Carlo (MC) event simulation techniques to provide a contrast between our work and current practices followed by the particle physics community. More detailed information about the LHC, the CMS experiment, as well as other experiments at the LHC can be found in [24, 25].

[1] Important caveat: only particles with masses lesser than the energy of collision will be created.

## 3.1 Relativistic Kinematics

Before we proceed to discuss elements of the LHC, let us take a slight detour to briefly describe the kinematics of particles produced at the LHC. Since protons in the proton beams and the resultant debris from the collision all travel very close to the speed of light, we cannot describe their motion merely with Newtonian mechanics. Objects travelling at speeds comparable to that of light are best described by the framework of special relativity, which operates in four-dimensional spacetime. Traditional three-vector positions in special relativity take the form of a four-vector:

$$x^\mu = (t, x, y, z) = (x_0, x_1, x_2, x_3) \tag{3.1}$$

The superscript $\mu$ can be lowered to obtain the dual of the position four-vector,

$$x_\mu = (t, -x, -y, -z) = (x_0, -x_1, -x_2, -x_3) \tag{3.2}$$

This allows us to define the dot product in Minkowski space (the space which special relativity describes) as

$$x^\mu y_\mu = y_\mu x^\mu = x_0 y_0 - x_1 y_1 - x_2 y_2 - x_3 y_3 \tag{3.3}$$

This dot product is a Lorentz-invariant quantity, which means that its value is independent of the frame of reference through we choose to study the system. Using the position, we can define the four-momentum as

$$p^\mu = m\frac{\mathrm{d}x^\mu}{\mathrm{d}\tau} \tag{3.4}$$

where $m$ is the object's rest mass and $\tau$ is the proper time described as

$$t = \gamma\tau, \quad \gamma = \frac{1}{\sqrt{1 - v^2}} \tag{3.5}$$

for the three-velocity $v = |\mathbf{v}|$.This gives us an equivalent expression for the momentum, defined as

$$p^\mu = (E, \mathbf{p}), \quad E = \gamma m, \mathbf{p} = \gamma m \mathbf{v} \tag{3.6}$$

If we take the Minkowski dot product of the momentum with itself, we obtain the Lorentz-invariant quantity

$$p^\mu p_\mu = E^2 - |\mathbf{p}|^2 \tag{3.7}$$

In the frame in which the object is at rest (for instance, its center-of-mass frame), $\mathbf{v} = 0 \implies p^\mu = (m, 0)$, and from Lorentz-invariance, it follows that

$$E^2 - |\mathbf{p}|^2 = m^2 \tag{3.8}$$

which is Einstein's energy-momentum relation, and the left-hand side is known as the invariant mass $s$, since it is equal to the rest mass of the object. As $\sqrt{s}$ is the actual rest mass, the

center-of-mass energy of the LHC is written as $\sqrt{s}$ =13.6 TeV.

For multiple particles, we can define an invariant mass for the system of objects with four-momenta $p_1, p_2, \ldots$ as

$$s_{1,2,\ldots} = \left( \sum_i p_1 \right)^2 \tag{3.9}$$

This quantity frequently appears in the search for new particles because it allows us to detect resonances, or intermediate particles the can produce different final states.

## 3.2 Large Hadron Collider

The LHC at the European Organization for Nuclear Research (CERN) is designed to accelerate two beams of protons to an energy of upto 7 TeV for collisions. Run 2 of the LHC operated at proton energies of 6.8 TeV resulting in a center-of-mass energy for proton-proton ($pp$) collisions of $\sqrt{s}$ = 13.6 TeV. These proton beams are allowed to collide at four specific interaction points where the four main experiments ALICE [26], ATLAS [27], CMS [25], and LHCb [28] are located. The ALICE detector is optimized for studying strong interactions via heavy-ion collisions, while the LHCb experiment focuses on B hadron physics. The ATLAS and CMS experiments are multi-purpose detectors that probe the SM in various ways and detect new particles.

Within the LHC synchrotron, superconducting dipole magnets are used to confine the proton beams to a circular trajectory through the Lorentz force. Particles with charge $q$ and momentum $\mathbf{p}$ moving in a magnetic field $\mathbf{B}$ follow the relation

$$p \propto qBR \tag{3.10}$$

where $R$ is the radius of the circular trajectory and $\mathbf{p} \perp \mathbf{B}$. Magnets with a field strength of upto 8 T are used to reach the target center-of-mass energy. Additional superconducting quadrupole magnets are installed to focus the proton beams and higher-order magnets are used for further corrections.

An important parameter used the describe the functioning of accelerators is the instantaneous luminosity $L_{\text{inst}}$ defined as

$$L_{\text{inst}} = \frac{f n_b N_b^2}{4\pi\sigma_x\sigma_y} \tag{3.11}$$

where $f$ is the revolution frequency of the proton bunches, $n_b$ is the number of proton bunches, $N_b$ is the number of protons per buch, and $\sigma_x$ and $\sigma_y$ descirbe the geometrical spread of the bunches perpendicular to their direction of motion. The luminosity provides a measure of the collision rate inside the accelerator. The total number of collisions for a given period of time

is proportional to the integrated luminosity $L_{\text{int}}$

$$L_{\text{int}} = \int dt\ L_{\text{inst}} \tag{3.12}$$

For example, the High-Luminosity Large Hadron Collider (HL-LHC) is expected to deliver an integrated luminosity of up to $3000\,\text{fb}^{-1}$ of high quality $pp$ collision data for physics analysis.

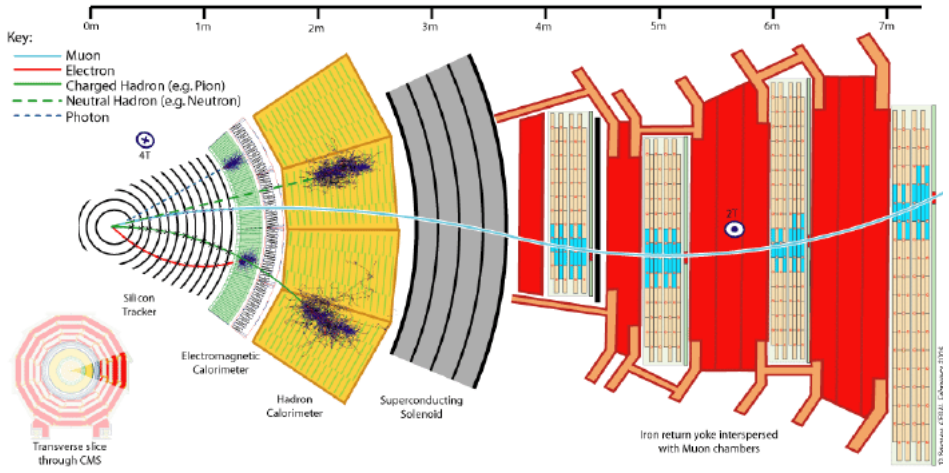


Figure 3.1: A schematic slice of the CMS detector in action. On the left is the innermost part closest to the beam, where protons collide. Followed by the tracker, the calorimeter systems, the solenoid magnet and the muon system to the far right. Additionally the detector response for different types of particles is illustrated. Taken from [29]

## 3.3 The CMS Experiment

The Compact Muon Soleniod (CMS) is one of two general-purpose detectors at the LHC. The goal of the CMS experiment is to validate the SM of particle physics and investigate beyond the standard model (BSM) phenomena. For this purpose, the CMS detector has to detect as many final-state particles as possible and precisely measure their properties. The CMS detector in constructed in a layerwise fashion, consisting of various subdetectors, each of which is designed for a different task. The subdetectors are divided into a barrel and two endcap regions.

The innermost layer of the CMS detector is the tracker which consists of silicon pixel and strip detectors which track the trajectories of all charged particles and their momenta. Followed by the tracker is the calorimetry system of the CMS detector which consists of an electromagnetic calorimeter (ECAL) and the hadronic calorimeter (HCAL) to measure the energy of the electromagnetic particles and hadrons, respectively. A superconducting solenoidal magnet surrounds the CMS detector and produces magnetic fields of 3.8 T. Muon chambers are embedded into the yoke of the magnet to measure the momenta of muons. These components are illustrated in Figure 3.1.

### 3.3.1 Coordinate system

The CMS detector follows a right-handed coordinate system centered around the interaction point. The $z$-axis lies along the beam axis in a counterclockwise direction. The $x$- and $y$-axes point to the center of the LHC ring and upwards to the surface, respectively. Given the cylindrical symmetry of the CMS detector, it is convenient to choose a cylindrical coordinate system to represent the detector elements. The polar angle $\theta$ is defined with respect to the $z$-axis, while the azimuthal angle is measured in the $x - y$ plane with respect to the $x$-axis. The radial distance $r$ is measured from the beamline.

While proton beams in the LHC have equal energies, the partons inside the protons can have different momenta along the beam axis. This results in a movement of the center-of-mass system along the $z$-axis relative to the laboratory frame, called boost. To counteract this, it is often reasonable to introduce Lorentz-invariant quantities such as the transverse momentum $p_T$ and rapidity $y$ to describe observables. These quantities are defined as follows

$$p_T = \sqrt{p_x^2 + p_y^2} \tag{3.13}$$

$$y = \frac{1}{2}\left(\frac{E + p_z}{E - p_z}\right) \tag{3.14}$$

Here, $E$ and $p_z$ are the energy and the $z$-component of the momentum of the particle, respectively. Differences of rapidity are invariant under Lorentz boosts. In the limit of high energies and negligible particle masses, it becomes equal to the pseudo-rapidity $\eta$ defined as

$$\eta = -\ln\left[\tan\left(\frac{\theta}{2}\right)\right] \tag{3.15}$$

depending only of the polar angle, which is directly measurable. The spatial distance between particles is measured by

$$\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2} \tag{3.16}$$

with $\Delta\eta = \eta_1 - \eta_2$ and $\Delta\phi = \phi_1 - \phi_2$.

The following subsections are heavily reproduced from [30].

### 3.3.2 The Tracker System

The tracker system of the CMS detector is located closest to the interaction point. This innermost layer is designed to precisely measure charged particles passing through with a pseudorapidity of $|\eta| < 2.50$. Charged particle are subjected to the Lorentz force due to the magnetic field which casues them to execute circular trajectories as defined in Equation 3.10. These particles interact with the tracker cells and leave energy deposits called “hits” in the tracker layer. Using these hits, the tracks of particles produced in the collision can be determined. With multiple tracks from one collision, it is possible to reconstruct the primary and

secondary vertices, which is useful to identiy the origin of some jets.

The tracker system is constructed using silicon to meet the high requirements in terms of radiation hardness, granularity and low material budget. The detector works similar to a diode in reverse bias. When charged particles hit the depletion region of the silicon, an electron-hole pair is generated. These charges are amplified, resulting in a current. The tracker consists of four inner layers of silicon pixel detectors with a resolution of approximately 10 µm to obtain a good vertex reconstruction despite high pileup. Layers of silicon strip detectors follow the pixel detector with increasing pitch of the strips as the particle flux decreases with radial distance.

### 3.3.3 The Electromagnetic Calorimeter

The Electromagnetic Calorimeter (ECAL) is designed for destructive energy measurements of electromagnetic particles like electrons, photons, and positrons. This calorimeter consists of lead tungstate ($PbWO_4$) crystal forming a homogeneous detector due to the high density, radiation hardness, and small radiation length of ($PbWO_4$).

Incoming photons with high energy primarily interact by producing electron-positron pairs ($\gamma \rightarrow e^- e^+$) in the field of a nucleus, whereas charged particles primarily lose their energies via brehmsstrahlung radiation ($e^- \rightarrow \gamma e^+$). These two processes produce secondary particles which interact in the same way with the detector material leading to an electromagnetic cascade, or an electromagnetic shower. The cascade continues until the energy of the shower is too low and charged particles ionize the detector material. Lead tungstate is an inorganic scintillator and converts the deposited energy into light which is measured by photodiodes. The energy deposition of the primary particle is proportional to the number of secondary particles, which in turn is proportional to the amount of emitted light.

### 3.3.4 The Hadronic Calorimeter

The Hadronic Calorimeter (HCAL) is built to measure the energy of particles interacting primarily through the strong force. It is a sampling calorimeter consisting of alternating layers of absorbers (brass and steel) with very high density and inorganic scintillators to convert the deposited energy into light. The interaction length of the HCAL is $\lambda = 16.42\,\text{cm}$.

The detection of hadron energy works similar to the ECAL with hadronic showers. The hadrons in fact interact with the ECAL, but the small interaction length of the ECAL leads to very little energy deposition of the hadrons. It should be noted that quarks are never directly observed in a detector. Instead, they rapidly decay into a spray of collimated hadrons (composite particles containing quarks) via the process of hadronization. These sprays are known as as jets, and the HCAL detects hadrons from these particle jets.

### 3.3.5 The Muon System

Identifying muons is an important aspect of multiple analyses. Most muons from proton-proton collisions at the LHC are so-called minimally ionizing particles (MIPs) which means they deposit only a small amount of energy in the detector material. Therefore the muons survive the destructive energy measurements of the calorimeters and can be detected in the outer muon system. The transverse momentum of these muons can be determined by measuring the bending of the muon trajectory in the 2 T magnetic field of the return yoke of the solenoid magnet and combining it with the tracker information.

### 3.3.6 The Trigger System

The LHC produces approximately 20 collisions per bunch crossing at a bunch crossing rate of 40 MHz. It is impossible to store all the data and most of the collisions are low energy QCD processes which are not the focus for CMS. To reduce the data rate feasible for storage a two-tier trigger system is used [31]. The Level-1 trigger (L1) is hardware-based and uses the calorimeter information and muon system to reduce the event rate to approximately 100 kHz. Followed by the second trigger called High-level trigger (HLT) using a large computer farm to perform a basic event reconstruction to further reduce the rate to several 100 Hz before the data is stored.

## 3.4 Monte Carlo Event Generation

Simulating particle physics events forms a bulk of the computational resource usage at the LHC. These simulations are used as a theoretical baseline against which experimental observations made in the CMS detector are compared. In high-energy physics (HEP), Monte Carlo (MC) simulations are powerful tools to calculate such theoretical predictions. MC is a numerical method based on random numbers, which scales well for high-dimensional problems.

Matrix Elements (ME) are mathematical objects that represent the probability of a certain process. To accurately simulate a proton-proton collision, all events need to be simulated at a parton level. First, MEs are perturbatively calculated with dedicated generators such as MadGraph5 [32]. A Parton Distribution Function (PDF) is chosen, then an integration over the final state phase space with MC is performed to obtain probability distributions. Last, a final state is sampled from these distributions.

After the ME calculations, the parton showers for the final state particles are simulated via a parton shower (PS) algorithms, like Pythia 8.2 [33]. These generators simulate gluon radiation and quark-antiquark pair production for the initial and final state partons.

The next step is the hadronization of the showers, which is a non-perturbative simulation relying on phenomenological models such as the cluster hadronization model or the Lund string

model. Next, the detector response is simulated using the GEANT4 software [1]. The output is comparable to detector data and the same reconstruction methods as for the detector can be used. To be able to produce these computationally expensive simulations centrally for different analyses and not to have to cover every single phase space, the generated events are weighted to match the observed data:

$$w_\sigma = \frac{\sigma_{\text{theo}} \cdot L_{\text{int}}}{N_{\text{gen}}} \tag{3.17}$$

$N_{\text{gen}}$ is the effective number of generated events, $\sigma_{\text{theo}}$ is the predicted cross section and $L_{\text{int}}$ is the integrated luminosity to which the expected numbers is scaled to.

In this thesis, the presented work is confined to the penultimate step of simulating the response of the CMS detector layers. Our model seeks to simulate the response of the the ECAL layer for boosted top quark events.

# Chapter 4

# Deep Learning

The idea of creating machines that "think" is almost as old as the history of modern computers. The idea of computational intelligence was first formalized in Alan Turing's seminal 1950s paper on Computing Machinery and Intelligence [34], but the development of connectionist models that imitate our brain goes back even further to McCulloch and Pitts [35] and Hebb's [36] implementation of the Perceptron. Although the popularity of neural networks briefly died down at the turn of the 21st century, they have undergone a significant and popular revival. The release of ImageNet in 2012, which successfully demonstrated how a neural network can be trained efficiently using graphics processing units (GPUs) created a new wave in the field. At the time of writing this thesis, neural networks are undergoing a burgeoning resurgence in popularity and interest as the tools of choice in machine learning. This popularity can be attributed to three major causes: first, the information age has heralded a deluge of available data to effectively train models, which has led to deep learning becoming more useful; second, improvements in software and hardware architectures have allowed more complex models to be trained with reasonable efficiency; and third, deep learning has been able to successfully solve complicated problems in almost all domains of science and engineering. In this thesis, we shall explore one such application of deep learning to particle physics.

In this chapter, we provide an overview of modern deep learning concepts and techniques, and also discuss what makes these models so successful in learning complex patterns from data.

## 4.1 Neural Networks

Perhaps the most popular of all machine learning models are neural networks — and for good reason: inspired by the brain, neural networks have been proven to be universal approximators, that is, given any continuous function $h(\boldsymbol{x})$, where $\boldsymbol{x}$ is an arbitrary real-valued vector, we can find a set of parameters $\theta$ such that a neural network $f(\boldsymbol{x};\theta)$ satisfies

$$|h(\boldsymbol{x}) - f(\boldsymbol{x};\theta)| < \epsilon \quad \forall \boldsymbol{x}, \text{for} \quad \epsilon > 0 \tag{4.1}$$

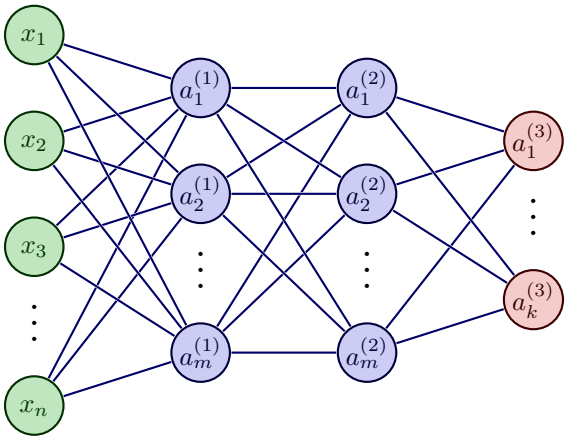


Figure 4.1: An example of a fully-connected feedforward neural network with two hidden layers. Each hidden layer consists of $m$ neurons, denoted by $a_j^{(l)}$, where $l$ denotes an index over the hidden layers and the index $j$ runs over the dimension of the hidden layer. This particular network accepts an $n$-dimensional input through the input layer (in green), passes the input through the hidden layers (in lavender), and produces a $k$-dimensional output at the output layer (in red). Solid lines between individual nodes denote weighted connections that are learned during the training procedure. Activation functions are not shown in this figure.

The task of "learning" involves finding this optimal set of parameters $\theta$, such that the above condition can be satisfied for many values of the input.

Multilayer Perceptrons (MLPs) are the simplest type of neural network architecture. A typical MLP consists of a number of processing units called neurons, arranged into one or many layers composed in a sequential fashion. A neuron performs a linear operation by aggregating weighted inputs received from neurons in the previous layer. Thus, each layer represents one round of this computation. An MLP generally consists of an input layer, followed by one or more hidden layers, and a final output layer consisting of one or more neurons. Computation in an MLP flows in a linear fashion, starting at the input layer and moving successively through the hidden layers until it reaches the output layer. Figure 4.1 provides an illustration of a simple MLP with one input layer, two hidden layers and an output layer.

At its core, an MLP operates by applying a set of weights $\boldsymbol{W}$ to the input, followed by a location transform $\boldsymbol{b}$. Finally, the MLP applies an activation function to explicitly introduce nonlinearity in the computational graph. Examples of some commonly used activation function are shown in Figure 4.2. The choice of activation function can depend on the use case, but ReLU activation functions (and their variants) have been popular choices for most applications.

### 4.1.1 A Matrix Formulation of MLPs

What contributes the most to the efficiency with which neural networks like MLPs can learn has to do with how their operations can be expressed as successive matrix-valued operations.

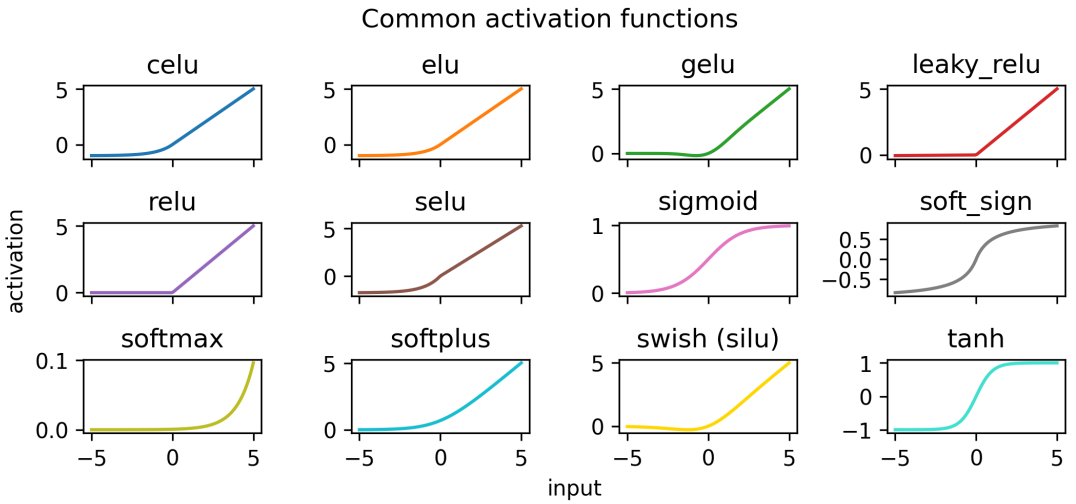


Figure 4.2: Some commonly used activation functions, taken from the Python library JAX.

Given a set of $n$ activations from the previous layer, and $k$ neurons in layer $i$, the $i+1$th activations are obtained as follows:

$$\begin{pmatrix} a_0^{i+1)} \\ a_1^{i+1)} \\ \vdots \\ a_n^{i+1)} \end{pmatrix} = \sigma \left[ \begin{pmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{pmatrix} \begin{pmatrix} a_0^{(i)} \\ a_1^{(i)} \\ \vdots \\ a_n^{(i)} \end{pmatrix} + \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{pmatrix} \right] \tag{4.2}$$

where the activation function $\sigma(\cdot)$ is applied elementwise. If we represent the activations, weights and biases as real-valued matrices, then we can succinctly write Eq. (4.2) as $\boldsymbol{a}^{(i+1)} = \sigma(\boldsymbol{W}\boldsymbol{a}^{(i)} + \boldsymbol{b})$ Thus, the parameters $\theta$ defined earlier typically represent the weights and biases associated with the neural network.

## 4.2 Gradient Descent

So far we have discussed what neural networks look like and how they are constructed, and we have stated that a neural network is parameterized by a set of weights that are "learnt". However, we have not yet uncovered how a neural network arrives at an optimum choice for its parameters — this job is accomplished by a learning algorithm. In this section, we'll dive into a very popular learning procedure called gradient descent and demonstrate how it helps us reach an optimum for a function. We'll also cover some contemporary modifications to this algorithm that are more commonly used in deep learning community.

Gradient descent is based on the assumption that for a given differentiable (multivariable) function $F(\boldsymbol{x})$, $F(\boldsymbol{x})$ decreases the most in the direction opposite to its gradient. Recall that the gradient of a function points in the direction of steepest increase, and thus moving in a

direction *opposite* to the gradient should eventually lead us to a minimum. Mathematically, if

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \eta \boldsymbol{\nabla} F(\boldsymbol{x}_n) \tag{4.3}$$

for a small enough learning rate $\eta \in \mathbb{R}^+$, then $F(\boldsymbol{x}_n) > F(\boldsymbol{x_{n+1}})$.

Let us explain this with an example. For simplicity, consider the function $f(x_1, x_2) = \frac{x_1^2+x_2^2}{4000} - \cos x_1 \cos\left(\frac{x_2}{\sqrt{2}}\right) + 1$. From its graph in Figure 4.3, we see that the function has a minimum at $(0, 0)$. Using a learning rate $\eta = 0.5$, we are able to converge to the miniumum in just a few iterations. The path traced out by the algorithm is indicated in Figure 4.3 by the red arrows. The learning rate indicates the "jump" the algorithm makes to find the next point in the sequence from Eq. (4.3).

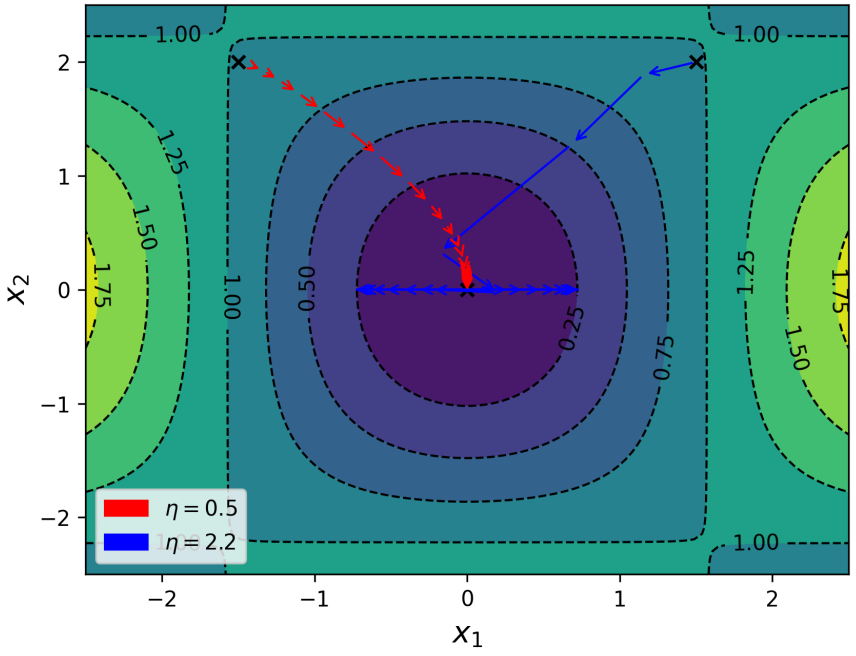


Figure 4.3: A plot of the multivariable function $f(x_1, x_2) = \frac{x_1^2+x_2^2}{4000} - \cos x_1 \cos\left(\frac{x_2}{\sqrt{2}}\right) + 1$, with two runs of gradient descent with learning rates $\eta = 0.5$ (red) and $\eta = 2.1$ (blue) to find the minimum at $(0, 0)$.

The choice of the learning rate is absolutely essential in deciding the effectiveness of gradient descent. If we choose a learning rate that is too small, it might take may more iterations than required to converge to the optimum point; on the other hand, if we choose a large learning rate, we might never be able to converge to an optimum, despite the function having a well-defined minimum. This situation is indicated by the blue run in Figure 4.3. By choosing a large learning rate, the algorithm reaches the vicinity of the minimum, but eventually becomes trapped in the well around it (defined by the contour at $f(x_1, x_2) = 0.25$). In the example above, each successive iteration sends the algorithm to the opposite side of the well, off-shooting the minimum, thus trapping it.

Now that we have a basic understanding of gradient descent, we can move on to discuss how

this algorithm is used in deep learning. Earlier, we mathematically defined a neural network to be a function $f(\boldsymbol{x};\theta)$, parameterized by a set of learnable parameters $\theta$. During training, the output from the network is compared against training data using a function called a loss function, $L$. Given a loss function $L(\theta, d)$ that defines a goal relative to (training) data $d$, we can use gradient descent to update the parameters $\theta$ such that we minimize the objective evaluated at $d$:

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L(\theta_i, d)}{\partial \theta_i} \tag{4.4}$$

An important caveat is that this method works only if the loss function is differentiable with respect to the parameters, otherwise the gradient cannot be computed.

## 4.3 Improving gradient descent

Although gradient descent is a powerful optimization tool, it can fall short of the requirements imposed by current deep learning architectures. In this section, we will cover some state-of-the-art optimization schemes that improve upon gradient descent to speed up convergence.

### 4.3.1 Momentum

If gradient descent is stuck in a steep valley around a local minimum, it typically bounces between the walls of the valley. However, if we were to introduce information about the previous update step, we can define an update rule with a placeholder variable $v$ like so:

$$v_i = \gamma v_{i-1} + \eta \frac{\partial L(\theta_i, d)}{\partial \theta_i}; \quad \theta_{i+1} = \theta_i - v_i \tag{4.5}$$

In this situation, we are able to speed up the update by an amount $\gamma v_{i-1}$, so if we performed a large update in the previous time step, we can go even further this time and vice versa. This technique is called gradient descent with momentum, analogous to the concept of mechanical momentum, hence the name.

### 4.3.2 Adagrad

In reality, most modern neural networks contains tens of thousands, if not millions of parameters. Thus, it would be unwise to update all the parameters in the neural network by the same amount at each optimization step. This is because certain parameters might be more influential in the decision process than the others, and therefore need sensitive updates. In other cases, some parameters might be logically flat in one area of the loss landscape and more influential in others. This gives rise to the notion of adaptive learning rates on a per-parameter basis to allow us to explore the loss landscape more effectively.

Adagrad (or "adaptive gradient") scales the learning rate for each parameter in such a way as to increase the step size for parameters that have moved less, and decrease it for the parameters that have undergone relatively larger updates. This is done by scaling the learning rate by an inverse of the sum of the gradients, $g$ from all previous steps. Mathematically, a parameter $\theta_i$ undergoes an update

$$\theta_{i+1} = \theta_i - \frac{\eta}{\sqrt{\sum_{j=0}^{i} g_j^2 + \epsilon}} \frac{\partial L(\theta_i, d)}{\partial \theta_i} \tag{4.6}$$

with the $\epsilon$ added for numerical stability, but this not enough. In many cases, the sum of the gradients itself can increase without bounds (called gradient explosion), which can bring the optimization to a screeching halt.

### 4.3.3 RMSProp

This time, instead of maintaining a sum over the squared gradients, we can recursively define a decaying average to remember the previous gradients, much like momentum.

$$E[g^2]_i = \gamma E[g^2]_{i-1} + (1 - \gamma) g_i^2 \tag{4.7}$$

The update rule can then be defined as

$$\theta_{i+1} = \theta_i - \frac{\eta}{\sqrt{\sum_{j=0}^{i} E[g^2]_j + \epsilon}} \frac{\partial L(\theta_i, d)}{\partial \theta_i} \tag{4.8}$$

### 4.3.4 Adam

We finally reach Adam [37], by far the most commonly used optimization algorithm in deep learning, as well as the one used in this thesis. Adam builds upon ideas from all of the previously defined methods: it maintains a decaying average of gradients and square gradients as:

$$m_i = \beta_1 m_{i-1} + (1 - \beta_1) g_i \tag{4.9}$$

$$v_i = \beta_2 v_{i-1} + (1 - \beta_2) g_i^2 \tag{4.10}$$

These quantities are actually Monte Carlo estimations of the first moment (mean) and the second moment (variance) of the gradients, $g$, hence the name Adam (adaptive moment estimation). The Adam authors note that by themselves, these terms can be biased towards zero,

and need a correction. This is done by applying bias correction terms to the moments

$$\hat{m}_i = \frac{m_i}{1 - \beta_1^i} \tag{4.11}$$

$$\hat{v}_i = \frac{v_i}{1 - \beta_2^i} \tag{4.12}$$

We then get the update rule

$$\theta_{i+1} = \theta_i - \eta \frac{\hat{m}_i}{\sqrt{\hat{v}_i} + \epsilon} \tag{4.13}$$

## 4.4 Generative Models

In this thesis, we want to simulate the detector response for a given type of particle physics event by *generating* detector hits for the event. Mathematically, we can interpret this simulation process as drawing samples from the multidimensional probability distribution of detector hits. To do this, we train a class of deep learning models called generative models, which, given some observed samples $\boldsymbol{x}$ from a distribution of interest, attempt to model the true underlying distribution $p(\boldsymbol{x})$. The study of generative models and developing the mathematical tools to understand and improve their behaviour is a vast and active area of research. In this section, we shall only provide a broad overview of the mathematical background required to understand the working of these models, followed by a description of variational methods to generate data.

### 4.4.1 Evidence Lower Bound

For most intents and purposes, we can think of data as being generated or represented by an associated latent variable, say $\boldsymbol{z}$. The data that we actually see may be generated as a function of such higher-level representations which encapsulate abstract properties such as colour, shape, size, etc. Thus, what we observe can be interpreted as a lower-dimensional projection of these abstract representations that are beyond our reach. Therefore, when we attempt to model these data distributions, we must attempt to approximate latent representations that describe the data we observe.

Mathematically, we can imagine the latent variables and the data we observe to be modelled by a joint distribution $p(\boldsymbol{x}, \boldsymbol{z})$. One goal of generative modelling is to maximize the likelihood $p(\boldsymbol{x})$ of all observed $\boldsymbol{x}$. We can obtain the likelihood for our data either by integrating over the latent variable

$$p(\boldsymbol{x}) = \int d\boldsymbol{z}\ p(\boldsymbol{x}, \boldsymbol{z}) \tag{4.14}$$

or, by applying the chain rule of probability:

$$p(\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} \tag{4.15}$$

Directly obtaining the likelihood and maximizing it is impractical because it either requires us to integrate over the entire space of latent variables is often computationally intractable for complex models, or involves having access to an encoder $p(\boldsymbol{z}|\boldsymbol{x})$. To circumvent this problem, we instead maximize a lower bound on the evidence, or the log-likelihood of the observed data. This quantity, called the **E**vidence **L**ower **BO**und, or **ELBO**, serves as a proxy objective with which to optimize a latent variable model. In the best case scenario, when the ELBO is parameterized and optimized perfectly, the ELBO is exactly equivalent to the evidence. The ELBO is defined as:

$$\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z},\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.16}$$

and,

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z},\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.17}$$

Here $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is a flexible approximate variational distribution with parameters $\phi$ that we seek to optimize. Intuitively, it can be thought of as a parameterized model that is learnt to estimate the true distribution over latent variables for a given observation $\boldsymbol{x}$; in other words, it seeks to approximate the true posterior, $p(\boldsymbol{z}|\boldsymbol{x})$.

**Theorem 1.** *The ELBO is a lower bound on the evidence* $\log p(\boldsymbol{x})$.

*Proof.*

$$\log p(\boldsymbol{x}) = \log p(\boldsymbol{x}) \int d\boldsymbol{z}\ q_\phi(\boldsymbol{z}|\boldsymbol{x}) \quad \left(\int d\boldsymbol{z}\ q_\phi(\boldsymbol{z}|\boldsymbol{x}) = 1\right) \tag{4.18}$$

$$= \int d\boldsymbol{z}\ q_\phi(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x}) \tag{4.19}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x})] \tag{4.20}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.21}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})}\frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.22}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.23}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] + D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{x},\boldsymbol{z})) \tag{4.24}$$

$$\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.25}$$

□

From the above proof, we observe that the evidence is equal to the ELBO plus the Kullback-Leibler (KL) Divergence [1] between the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ and the true posterior

[1] The KL Divergence between two continuous distributions $P$ and $Q$ is defined as $D_{\mathrm{KL}}(P \parallel Q) = \int_{\mathbb{R}} \mathrm{d}x\ p(x) \log\left(\frac{p(x)}{q(x)}\right)$

$p(\boldsymbol{z}|\boldsymbol{x})$. We want to optimize the parameters of our variational posterior to exactly match the true posterior distribution, which is achieved by minimizing their KL Divergence (ideally to zero). Unfortunately, it is intractable to minimize this KL Divergence term directly, as we do not have access to the ground truth $p(\boldsymbol{z}|\boldsymbol{x})$ distribution. However, the likelihood of our data (and therefore the evidence term) is always a constant with respect to $\phi$, as it is computed by marginalizing out all latents $\boldsymbol{z}$ from the joint distribution $p(\boldsymbol{z}, \boldsymbol{x})$ and does not depend on $\phi$ at all. Since the ELBO and KL Divergence terms sum up to a constant, any maximization of the ELBO term with respect to $\phi$ necessarily invokes an equal minimization of the KL Divergence term. Thus, the ELBO can be maximized as a proxy for learning how to perfectly model the true latent posterior distribution; the more we optimize the ELBO, the closer our approximate posterior gets to the true posterior.

### 4.4.2 Variational Autoencoders

Variational Autoencoders (VAE) [38], are the simplest generative model architectures, which directly maximize the ELBO. The name variational comes from how they optimize for the best approximate posterior amongst a family of potential posterior distributions paramterized by $\phi$. The term autoencoder refers to the physical architecture of the mode, which is reminiscent of a traditional bottleneck autoencoder, where the input data is trained to predict itself after passing through an intermediate bottleneck. To make this connection explicit, let us revisit the ELBO term:

$$\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.26}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{4.27}$$

$$= \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]}_{\text{reconstruction term}} - \underbrace{D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z}))}_{\text{prior matching term}} \tag{4.28}$$

In a VAE, we learn an intermediate bottlenecking distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ that is called the "encoder", because it *encodes* the input data by mapping it to latent space; simultaneously, we learn a deterministic function $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ called the "decoder" to convert a given latent vector $\boldsymbol{z}$ into an observation $\boldsymbol{x}$.

The two terms in Eq. (4.28) each have intuitive descriptions: the first term measures the reconstruction likelihood of the decoder from our variational distribution; this ensures that the learned distribution is modelling effective latents that the original data can be regenerated from. The second term measures how similar the learned posterior distribution is to a prior belief held over latent variables. Minimizing this term encourages the encoder to actually learn a distribution rather than collapse into a Dirac delta function. Maximizing the ELBO is thus equivalent to maximizing its first term and minimizing its second term. Therefore, a VAE optimized the ELBO jointly over parameters $\phi$ and $\theta$. The encoder of the VAE is commonly chosen to model a multivariate Gaussian with diagonal covariance, and the prior is of-

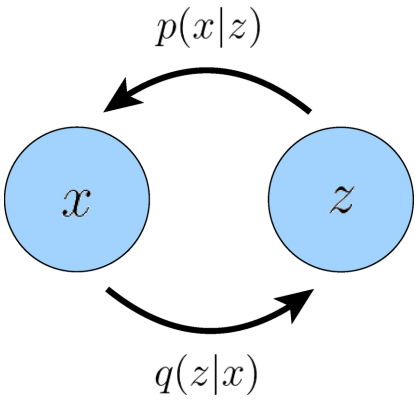


Figure 4.4: A graphical representation of a Variational Autoencoder. Here, encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ defines a distribution over latent variables $\boldsymbol{z}$ for observations $\boldsymbol{x}$, and decoder $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ decodes latent variables into observations [39].

ten selected to be a standard multivariate Gaussian:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_\phi(\boldsymbol{x}), \boldsymbol{\sigma}^2_\phi(\boldsymbol{x})\boldsymbol{I}) \tag{4.29}$$

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I}) \tag{4.30}$$

This allows us to obtain closed form expressions for the KL-divergence term, and the reconstruction term can be computed using a Monte Carlo estimate. After training a VAE, generating new data can be performed by sampling directly from the latent space $p(\boldsymbol{z})$ and then running it through the decoder.

# Chapter 5

# Score-based Generative Modelling

Deep generative models have been shown to produce high quality and high fidelity samples across a wide range of domains [40]. Generative adversarial networks (GANs) [41] remain the current preferred choice for sample generation due to their better sample quality over likelihood-based models like variational autoencoders [38] or normalizing flows [42]. However, GANs impose restrictions over their architecture and optimization strategies in order to stabilize training [43, 44], and also have issues covering the entire data distribution [45].

Recently, iterative generative models such as diffusion-based models [46] have demonstrated the ability to produce samples comparable to GANs without the need for adversarial training strategies. Instead, diffusion-based models successively apply Gaussian noise to data samples, and learn to denoise samples at each step. Samples are then generated from white noise by a Markov chain which progressively denoises it to produce a data instance. This Markov chain is either based on Langevin dynamics, or obtained by reversing a forward diffusion process.

## 5.1 Denoising Diffusion Probabilistic Models

First introduced by Ho et al. [46], Denoising Diffusion Probabilistic Models (DDPMs) are a class of Markov chain-based iterative generative models that are trained using variational inference. Transitions of this Markov chain are learned to reverse a diffusion process, which is a Markov chain that gradually adds noise to the data in the opposite direction of sampling until signal is destroyed.

Consider a data point $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$. In a diffusion model, we learn a reverse process is the distribution $p_\theta(\boldsymbol{x}) = \int d\boldsymbol{x}_{1:T}\ p_\theta(\boldsymbol{x}_{0:T})$, where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ are latents of the same dimensionality as $\boldsymbol{x}_0$. The reverse process is implemented as a Markov chain with $T$ timesteps and with learned

Gaussian transitions starting at $p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x_T}; \mathbf{0}, \boldsymbol{I})$:

$$p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T) \prod_{t=t}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t), \quad p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x_{t-1}}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)) \tag{5.1}$$

The posterior $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_T)$, which is called the forward process is also a fixed Markov chain that gradually adds noise to the initial data point $\boldsymbol{x}_0$ over $T$ time steps using a variance schedule $\{\beta_1, \ldots, \beta_T\}$:

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=t}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \quad q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x_t}; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{5.2}$$

Training is performed by optimizing the ELBO on the negative log likelihood:

$$\mathbb{E}[-\log p_\theta(\boldsymbol{x}_0)] \leq \mathbb{E}_q \left[ -\log \frac{p_\theta(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \tag{5.3}$$

$$= \mathbb{E}_q \left[ -\log p(\boldsymbol{x}_T) - \sum_{t \geq 1} \log \frac{p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right] \tag{5.4}$$

The forward process variances $\beta_t$ can either be learnt via reparameterization or held constant as hyperparameters. The functional form of the forward process transition kernels admits sampling $\boldsymbol{x}_t$ at an arbitrary timestep $t$ in closed form:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x_t}; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I}) \tag{5.5}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha} = \prod s = 1^t \alpha_s$. The training procedure can be further optimized by rewriting the loss as:

$$\mathbb{E}_q \left[ \underbrace{D_{KL}(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \parallel p(\boldsymbol{x}_T))}_{L_T} + \underbrace{\sum_{t>1} D_{KL}(q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) \parallel p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))}_{L_{t-1}} - \underbrace{\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right] \tag{5.6}$$

Eq (5.6) uses KL divergence to directly compare $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ against forward process posteriors, which are tractable when conditioned on $\boldsymbol{x}_0$:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x_{t-1}}; \tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t, \boldsymbol{x}_0), \beta_t \boldsymbol{I}), \tag{5.7}$$

$$\text{where} \quad \tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\boldsymbol{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t \tag{5.8}$$

Consequently, all KL divergences in Eq (5.6) are comparisons between Gaussians, so they can be computed with closed form expressions instead of Monte Carlo estimates.

Given the choice of our kernels, with $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x_{t-1}}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \sigma^2 \boldsymbol{I})$ [here, $\sigma_t^2 = \beta_t$], we

can write:

$$L_t = \mathbb{E}\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)\|^2\right] + C \tag{5.9}$$

So, the most straightforward parameterization of $\boldsymbol{\mu}_\theta$ is a model that predicts $\tilde{\boldsymbol{\mu}}$, the forward process posterior mean. However, we can expand Eq. (5.9) further by reparameterizing $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$ for $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Ultimately, we can simplify the training objective even further by choosing the following parameterization:

$$\boldsymbol{\mu}_\theta = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right) \tag{5.10}$$

where $\epsilon_\theta(\boldsymbol{x}_t, t)$ is a function approximation that predicts the noise at each time step in the forward process. To sample $\boldsymbol{x}_{t-1} \sim p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is to compute

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right) + \sigma_t \boldsymbol{z}, \quad \text{where} \quad \boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{5.11}$$

Furthermore, with this chosen parameterization, the loss to optimize simplifies to

$$\mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right] \tag{5.12}$$

which resembles denoising score matching over multiple noise scales indexed by $t$.

As it turns out, the term $-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ represents the "score" function, which is defined as the gradient of the log-likelihood with respect to the data itself, or $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, and we might choose the score parameterization to instead predict the score at each time step of the Markov process.

## 5.2 Score-based Generative Modelling with SDEs

In the previous section, we observed that the key component of the entire process is to perturb data with multiple noise scales. This idea can be generalized further to an infinite number of noise scales (time steps), such that the perturbed data distributions evolve according to a stochastic differential equation (SDE). Then, the discrete latent variables are replaced by a continuous latent function $\boldsymbol{x}(t)$. Our goal now is to construct a diffusion process $\{\boldsymbol{x}(t)\}_{t=0}^T$, index by a continuous time variable $t \in [0, T]$ such that $\boldsymbol{x}(0) \sim p_0$ and $\boldsymbol{x}(T) \sim p_T$, for which we have a computationally tractable form to generate samples. If $p_0$ is the data distribution and $p_T$ is the prior distribution, then the diffusion process can be modeled as the solution to an Itô SDE:

$$d\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)dt + g(t)d\boldsymbol{w} \tag{5.13}$$

where $\boldsymbol{w}$ is the standard Wiener process (also known as Brownian motion). Furthermore, $\boldsymbol{f}(\cdot, t) : \mathbb{R}^d \to \mathbb{R}^d$ is a vector-valued function called the drift coefficient of $\boldsymbol{x}(t)$ and $g(\cdot) : \mathbb{R} \to \mathbb{R}$ is a scalar-valued function known as the diffusion coefficient of $\boldsymbol{x}(t)$. For convenience,

we denote the probability density of $\boldsymbol{x}$ as $p_t(\boldsymbol{x})$ and the transition kernel from $\boldsymbol{x}(s)$ to $\boldsymbol{x}(t)$ as $p_{st}(\boldsymbol{x}(t)|\boldsymbol{x}(s))$, where $0 \leq s < t \leq T$. Like before, $p_T$ is an unstructured prior distribution which contains no information of $p_0$, and is typically chosen to be a standardized Gaussian distribution.

### 5.2.1 Generating Samples using the SDE

By starting from samples of $\boldsymbol{x}(T) \sim p_T$ and reversing the process, we can obtain samples $\boldsymbol{x}(0) \sim p_0$. As it turns out, the reverse of a diffusion process is also a diffusion process, running backwards in time and given by the reverse time SDE:

$$d\boldsymbol{x} = [\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2 \nabla_{\boldsymbol{x}} p_t(\boldsymbol{x})]dt + g(t)d\bar{\boldsymbol{w}} \tag{5.14}$$

where $\bar{\boldsymbol{w}}$ is a standard Wiener process when time flows backwards from $T$ to 0, and $dt$ is an infinitesimal negative time step. Once the score of each marginal distribution, $\nabla_{\boldsymbol{x}} p_t(\boldsymbol{x})$ is known for all $t$, we can derive the reverse diffusion process and simulate it to obtain samples $\boldsymbol{x}(0) \sim p_T$.

### 5.2.2 Estimating Scores for the SDE

The score of a distribution can be estimated by training a score-based model on samples with score-matching, as proposed in [47, 48]. This can be achieved by training a time-dependent score-based model $s_\theta(\boldsymbol{x}, t)$ for the following loss function:

$$\mathbb{E}_{\boldsymbol{x}(t)|\boldsymbol{x}(0)} \left[ \left\| \boldsymbol{s}_\theta(\boldsymbol{x}(t), t) - \nabla_{\boldsymbol{x}(t)} \log p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0)) \right\|^2 \right] \tag{5.15}$$

where $t \sim U(0, T)$, $\boldsymbol{x}(0) \sim p_0(\boldsymbol{x})$ and $\boldsymbol{x}(t) \sim p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$. With sufficient data and model capacity, score matching ensures that the optimal solution $\boldsymbol{s}^*_\theta(\boldsymbol{x}(t), t)$ equals $\nabla_{\boldsymbol{x}(t)} \log p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$ for almost all $\boldsymbol{x}$ and $t$. We also typically need to know the transition kernel $p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$, but the choice of kernel is fixed to a Gaussian with closed form expressions for the mean and variance.

For all diffusion processes, there exists a corresponding deterministic process whose trajectories share the same marginal probability densities $\{p_t(\boldsymbol{x}(t))\}_{t=0}^T$ as the SDE. This deterministic process satisfies an ODE:

$$d\boldsymbol{x} = \left[ \boldsymbol{f}(\boldsymbol{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) \right] dt \tag{5.16}$$

which can be determined from the SDE once the scores are known. This ODE is called the probability flow ODE. If the score function is approximated by a neural network, then this ODE takes the form of a neural ODE [49].

Recall the discrete variance-preserving [1] formulation we introduced for DDPMs in Section 5.1. Eq (5.10) applies to a *noise prediction* model, where we predict the noise that is added at each time step. Alternatively, we can instead formulate the DDPM as either a *denoising model*, to predict the original data point $\boldsymbol{x}_\theta(\boldsymbol{x}_t, t)$ or as a score model $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$, to predict the gradient of the log-likelihood at each step — all three views are equally valid. The three formulations are interrelated as:

$$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{x}_t - \bar{\alpha}_t \boldsymbol{x}_\theta(\boldsymbol{x}_t, t)}{\sigma_t} \tag{5.17}$$

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) = \frac{\bar{\alpha}_t \boldsymbol{x}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{x}_t}{\sigma_t^2} \tag{5.18}$$

As discussed in Section 5.2.2, when $T \to \infty$, we can describe the diffusion process through a probability flow ODE. Following Kingma et al [50], we obtain the following closed-form expressions for the drift and diffusion coefficients:

$$f(\boldsymbol{x}_t, t) = \frac{\mathrm{d} \log \bar{\alpha}_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma^2(t)}{\mathrm{d}t} - 2\frac{\mathrm{d} \log \bar{\alpha}_t}{\mathrm{d}t}\sigma^2(t) \tag{5.19}$$

The ODE in Eq. (5.16) can be solved numerically using standard solvers like the Euler or the Runge-Kutta methods. However, the DDIM solver proposed in [51] is also an integration rule for this ODE, as shown in [52]. Using the DDIM sampler, the update rule (for a denoising model) is:

$$\boldsymbol{x}_{t-1} = \bar{\alpha}_{t-1} \boldsymbol{x}_\theta(\boldsymbol{x}_t, t) + \sigma_{t-1} \frac{\boldsymbol{x}_t - \bar{\alpha}_t \boldsymbol{x}_\theta(\boldsymbol{x}_t, t)}{\sigma_t} \tag{5.20}$$

Following [52], in this thesis, we train a model $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ to predict the velocity parameter $\boldsymbol{v}$ at each time step, defined as:

$$\boldsymbol{v} = \bar{\alpha}_t \boldsymbol{\epsilon} - \sigma_t \boldsymbol{x}_0, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{5.21}$$

This choice has been shown to stabilize training. During sampling, the predicted velocity is used to generate the original data sample using the equation below, which is then used to predict the mean at each step, as indicated in Eq. (5.8).

$$\hat{\boldsymbol{x}}_0 = \bar{\alpha}_t \boldsymbol{x}_t - \sigma_t \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) \tag{5.22}$$

In this formulation, the score can be derived from the velocity as:

$$\nabla_{\boldsymbol{x}} p_\theta(\boldsymbol{x}_t) = \frac{\boldsymbol{x}_t(\bar{\alpha}_t^2 - 1)}{\sigma_t^2} - \frac{\bar{\alpha}_t}{\sigma_t} \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) \tag{5.23}$$

[1] By variance-preserving, we mean that at each step in the Markov chain, the scale of the variances used in the transition kernels remains the same.

Moreover, the loss function simplifies to:

$$L = \mathbb{E}_{\epsilon,t}\left[\left\|\boldsymbol{v} - \boldsymbol{v}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right] \tag{5.24}$$

The training and sampling algorithms are summarized below:

**Algorithm 1** Training

**repeat**
  $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$
  $t \sim U(0,1)$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{v} \leftarrow \bar{\alpha}_t\boldsymbol{\epsilon} - \sigma_t\boldsymbol{x}_0$
  $\hat{\boldsymbol{v}} \leftarrow \boldsymbol{v}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)$
  Take gradient descent step on:
    $\nabla_\theta\|\boldsymbol{v} - \hat{\boldsymbol{v}}\|^2$
**until** converged

**Algorithm 2** Sampling

$\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
**for** $t = T, \ldots, 1$ **do**
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\hat{\boldsymbol{v}} \leftarrow \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$
  $\hat{\boldsymbol{x}}_0 \leftarrow \bar{\alpha}_t\boldsymbol{x}_t - \sigma_t\hat{\boldsymbol{v}}$
  $\boldsymbol{\mu}_t \leftarrow \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\boldsymbol{x}}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t$
  $\Sigma_t = \beta_t\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$
  $\boldsymbol{x}_{t-1} = \boldsymbol{\mu}_t + \Sigma_t\boldsymbol{\epsilon}$
**end for**
**return** $\boldsymbol{x}_0$

# Chapter 6

# Methods and Results

## 6.1 Data

### 6.1.1 Open Data Simulated Samples

For this study, we are concerned with simulating the calorimeter shower distribution for $t\bar{t}$ events as measured by the CMS experiment at the LHC. We use a sample of SM top-antitop pair production where the W boson from the top quark decay is required to decay to quarks as a source of boosted top quarks. A snippet of a Feynman diagram corresponding to this process is shown below in Figure 6.1. The full dataset can be found in [53]. For all samples, the calorimeter and detector response is simulated using GEANT4 with full CMS geometry and processed through the CMS Particle Flow algorithm. This dataset includes low-level detector information, specifically reconstructed clusters from pixel and silicon strip detectors.

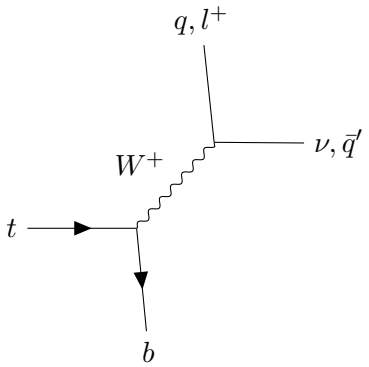


Figure 6.1: Decay of a top quark to a W boson and a bottom quark. The W boson then decays to two other quarks, making a triplet of jets. [54]

To form jets from low-level physics objects, we cluster using the anti-$k_T$ clustering algorithm [55] with a radius parameter $R = 0.8$ and limit the clustering to events with a pseudorapidity $|\eta| < 1.57$. This cut ensures that the jet does not extend beyond the $|\eta| < 2.4$ threshold

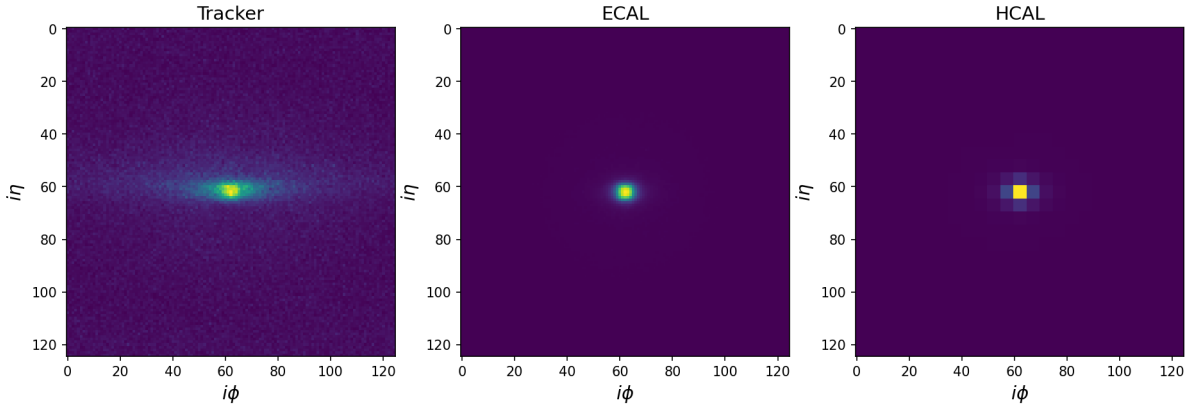


Figure 6.2: Jet image overlays split by subdetector: tracks, ECAL, and HCAL averaged over 70,000 jets each. Image resolution: $125 \times 125$.

for the current CMS tracker. We also require that the generator-level top quark, its bottom quark, and the W boson daughters to be within $\Delta R < 0.8$ from the reconstructed jets axis.

### 6.1.2 CMS Detector and Data

As discussed previously in Chapter 3, the CMS detector consists of a tracker layer followed by a calorimeter layer divided into the electromagnetic and hadronic calorimeters. CMS Open Data contains reconstructed hits for the ECAL and HCAL. Using imaging techniques from [56, 57], we are able to obtain calorimeter images whose pixels correspond to the calorimeter cells. Thus, full detector images consist of three subdetector channels: one each for ECAL, HCAL, and the Tracker.

From the multichannel image described above, we localize jet by taking the centroid of the reconstructed jet and then scan the HCAL channel for the highest energy deposit within a $9 \times 9$ window. Around this pixel, we crop a window of size $125 \times 125$ (corresponding to $\Delta R < 1$). This imposes an effective pseudorapidity cut of $|\eta| < 1.57$.

Figure 6.2 shows the various sub-detector image overlays averaged over the full test set of about 70,000 jets, while Figure 6.3 shows subdetector images for a single jet. The difference in detector resolution between the ECAL and HCAL layer is clearly visible, with HCAL deposits appearing "blockier" than ECAL deposits. The end-to-end images appear noticeably more "raw" in that they contain more noise and stray hits. This is expected given that we are looking at the physical detector deposits in all their richness.

### 6.1.3 Point Cloud Representation

Over the past decade, there have been extensive studies demonstrating state-of-the-art generative models for image datasets [56, 57, 58, 59, 60], but a common feature across all such

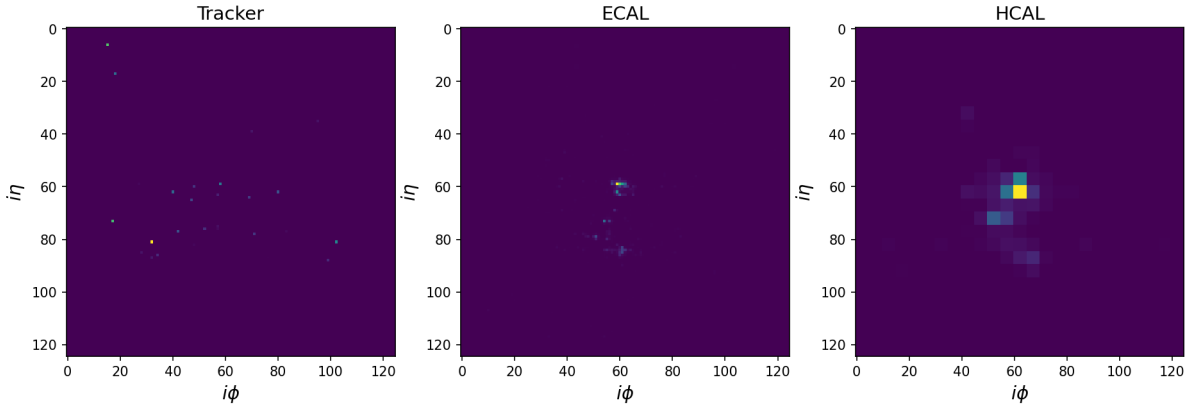


Figure 6.3: Jet image for a single jet split by subdetector: tracks, ECAL, and HCAL. Image resolution: $125 \times 125$.

studies is that the datasets all contain standard three-channel images of real-world objects (excluding, of course, studies with the MNIST dataset). Image-based representations of detector deposits, such as those described above, involve projecting the spatial data onto a discretized $\eta - \phi$ plane, and taking the intensity of each pixel in the image to represent the energy deposited in the calorimeter crystal or cell. However, such representations tend to be extremely sparse, with typically fewer than 10% of all pixels being nonzero. In this case, a loss function would compare a model's output to mostly zeros, and a model trained on sparse data can learn to output too many zeros, affecting its performance. Moreover, such a representation overlooks complex detector granularity and inhomogeneous detector resolution across layers. This is especially true in the regions that overlap between the barrel and endcap regions of the CMS detector, where a fixed-size $(\eta, \phi)$ resolution of $(0.025 \times 0.025)$ heavily undermines the complex geometry in this region. This mismatch in resolution also poses difficulties in the construction of high-fidelity detector projection images. Anecdotally, we also observed that while such representations were definitely helpful to train discriminative models like top-taggers, generative models struggled to faithfully learn the distribution of hit energies and generate high-quality samples of detector responses.

In the context of machine learning (ML), detector deposits have alternative representations in the form of either as ordered lists or unordered sets. The problem with the former is that there is no preferred natural ordering of the deposits – to implement such a representation, we will have to impose some nonphysical ordering on the constituents. A more natural representation of detector hits is the unordered set of hits in position space, which we call a point cloud.

To obtain a point cloud representation for an event, we start with the image-based representation of the jet shower that we obtained using the procedure described in the previous section. For each jet image, beginning from the top left corner we assign a coordinate $(x, y, z)$ to each pixel, where $0 \leq x \leq 124$ and $0 \leq y \leq 124$ are its position within the image, and $z$ is the pixel intensity. Then, we map each pixel to the corresponding coordinate in 3D space

to obtain a particle cloud for the event. For ease of batching during training, we must ensure that each event contains an identical number of hits. This can either be done by choosing to retain only $N$ hits in each event file, or by choosing a number $N$ and zero padding all events with less than $N$ hits in them. We choose to adopt the first method for convenience, as the second method requires us to learn and maintain a mask throughout the training and generation process which can be cumbersome. For this work, we choose to retain the top $N = 300$ hits in each event ordered by hit energy.

In our work, we presently simulate only the ECAL channel. Although a complete generative model would be capable of simultaneously simulating all three channels, limiting our discussion to just one channel allows us to easily optimize the model and makes computation more tractable. In the future, we plan to gradually add the functionality to simulate the remaining channels.

### 6.1.4 Preprocessing

Prior to training, all inputs to the model undergo a three-step normalization process. First, we apply a min-max transformation on all three features to map the inputs to the $[0, 1]$ range using the formula

$$\boldsymbol{x}_{\text{scaled}} = \frac{\boldsymbol{x} - \boldsymbol{x}_{\text{min}}}{\boldsymbol{x}_{\text{max}} - \boldsymbol{x}_{\text{min}}} \tag{6.1}$$

This is followed by a logit transformation, which transforms the features to log-space. The log-transformed features are defined as:

$$\text{logit}(\boldsymbol{z}) = \log\left(\frac{\boldsymbol{z}}{1 - \boldsymbol{z}}\right) \tag{6.2}$$

Finally, we standardize the inputs by adjusting their mean and variance to 0 and 1, respectively.

## 6.2 Model Architecture

For the forward process, we set the number of time steps to $T = 512$ and use the cosine variance schedule (in terms of $\bar{\alpha}$) proposed in [61], defined as:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1 + s} \cdot \frac{\pi}{2}\right)^2 \tag{6.3}$$

We can then use this definition to derive the variances $\beta_t = 1 - \bar{\alpha}_t/\bar{\alpha}_{t-1}$. The variances are then clipped to a range of $[0, 0.999]$ to prevent unexpected behaviour at $t = T$. A small offset $s = 8 \times 10^{-3}$ is added to prevent the variance from vanishing close to $t = 0$.

For the reverse process, we use a DeepSets [62] backbone with added Transformer [63] lay-

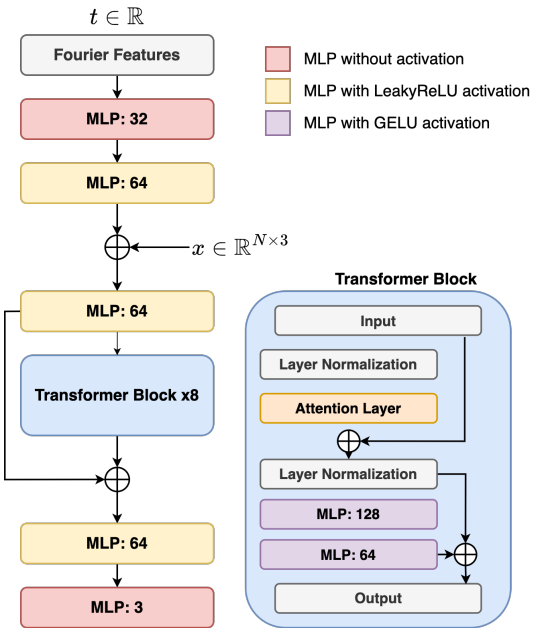


Figure 6.4: A visual description of the backbone architecture used in the diffusion model. Numbers after the layers represent the dimensionality of the layer output.

ers containing self-attention to augment the model's spatial expressivity. The input points are first mapped to a larger latent space with a dimension of 64. Then, the model uses eight Transformer blocks followed by a fully-connected layer before the output. A LeakyReLU activation function is used, and the outputs of the Transformer layers are summed to the last later before the first Transformer block, which is observed to result in better performance. This backbone is identical to the particle model architecture proposed in [64]. A visual description of the model is shown in Figure 6.4.

Time-domain information is incorporated in the network by passing random Fourier features [65] through two fully connected networks with 32 and 64 hidden nodes. The resulting time embedding is then concatenated with the hit features.

The model is implemented in PyTorch [66]. The model is trained for upto 200 epochs with a cosine learning rate schedule with an initial learning rate of $1 \times 10^{-3}$. Training is performed on a single NVIDIA DGX A100 GPU, with a batch size of 128.

## 6.3 Results

The performance of the diffusion model is evaluated using the 1-Wasserstein distance ($W_1$) and the Earth Mover's Distance (EMD) metrics. The EMD between two distributions is a similarity measure that quantifies the distance it takes to transform one distribution into the other. In the context of particle clouds, it represents the minimum work needed to be applies by the movements of energy $f_{pq}$ from particle $p$ in one cloud to particle $q$ in the other such

that the event $E$ is rearranged to event $E'$. For two clouds $P$ and $Q$, the EMD is mathematically defined as:

$$\mathrm{EMD}(P,Q) = \min_{f_{pq}} \sum_{p\in P}\sum_{q\in Q} f_{pq}\frac{R_{pq}}{R} + \left|\sum_{p\in P} s_{T_p} - \sum_{q\in Q} s_{T_q}\right|$$

subjected to the conditions

$$f_{pq} \geq 0 \tag{6.4}$$

$$\sum_{q\in Q} f_{pq} \leq s_{T_q},\ \sum_{p\in P} f_{pq} \leq s_{T_p} \tag{6.5}$$

$$\sum_{p\in P}\sum_{q\in Q} f_{pq} = \min\left(\sum_{p\in P} s_{T_p}, \sum_{q\in Q} s_{T_q}\right) \tag{6.6}$$

where $R$ is the radius of the jet event, and $R_{pq} = \sqrt{(y_p - y_p)^2 + (\phi_p - \phi_q)^2}$ is the distance in the rapidity-azimuth plane.

The $W_1$ metric is the one-dimensional analogue of the EMD, and for two distributions $u$ and $v$, is defined as:

$$W_1(u,v) = \inf_{\pi\in\Gamma(u,v)} \int_{\mathbb{R}\times\mathbb{R}} |x-y|\ \mathrm{d}\pi(x,y) \tag{6.7}$$

where $\Gamma(u,v)$ is the set of probability distributions on $\mathbb{R}\times\mathbb{R}$ whose marginals are $u$ and $v$ on the first and second factors, respectively. The EMD is calculated between two event point clouds, and the $W_1^{E_\mathrm{T}}$ distance is calculated between the total energies in each event. To calculate each metric, $20,000$ generated events are compared against 20,000 validation set events not used during training.

In Figure 6.5, we show some reconstruction samples for several simulated jets from the diffusion model along with randomly drawn samples from a held-out validation set. A visual inspection reveals that the model is able to successfully recreate jets from pure noise, both in terms of topological configuration as well as the energy scales of individual hits.

The total energy distribution is shown in Figure 6.6. The total energy for an event is computed by performing a sum over all hit energies in any given event. We observe a good agreement between GEANT4 and samples generated from the diffusion model. At the low energy fraction region, we see an excellent agreement with GEANT4, although there is a fairly trivial difference of almost 10% between the true and generated energies at both sides of the mean. This agreement continues until a total energy of $E_T = 250$ GeV, after which we see that the diffusion model fails to generate any contributions in the high energy fraction region with $E_T \geq 250$ GeV, leading to a sharp 100% difference between real and generated distributions at $E_T \approx 325$ GeV. This unexpected behaviour is uncharacteristic of diffusion models, which generally are able to faithfully model the target distribution with fairly high fidelity. Further investigations are needed to ascertain the cause of this behaviour. Quantitatively, the distri-

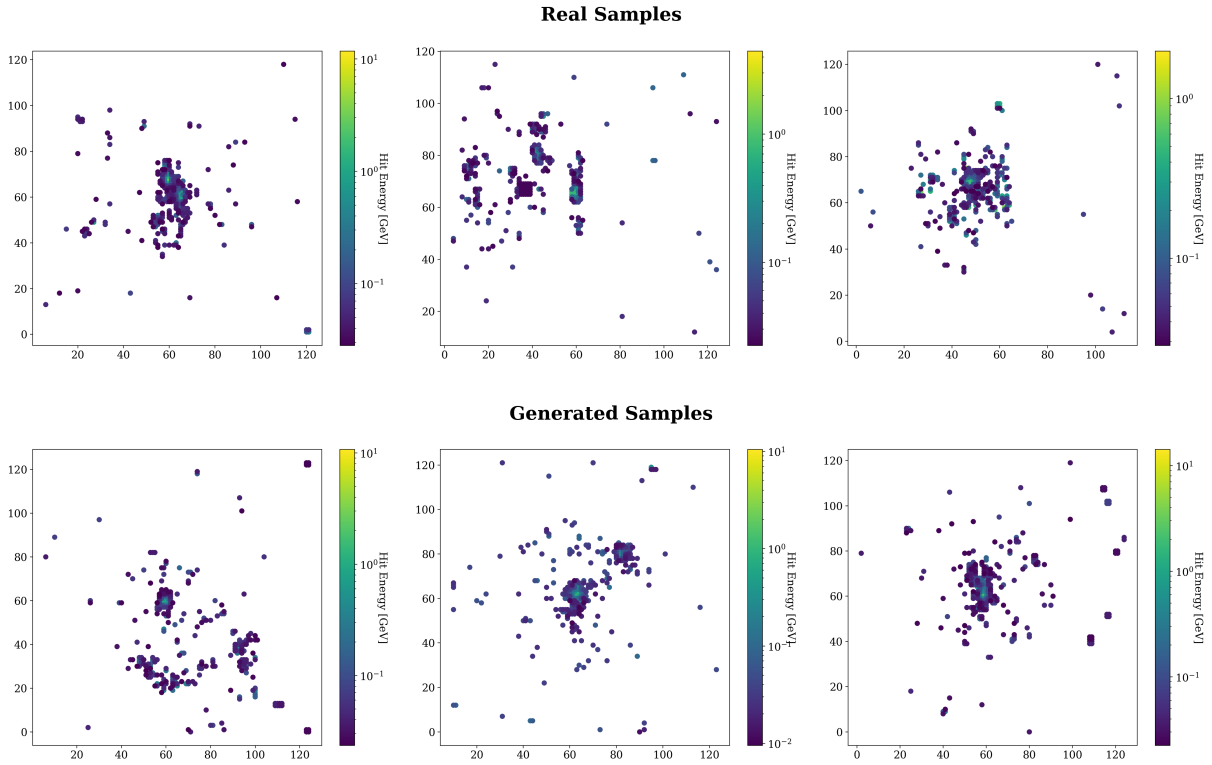


Figure 6.5: Real jet deposits (above) as compared to simulated jet deposits (bottom). Real samples were drawn randomly from a validation set.

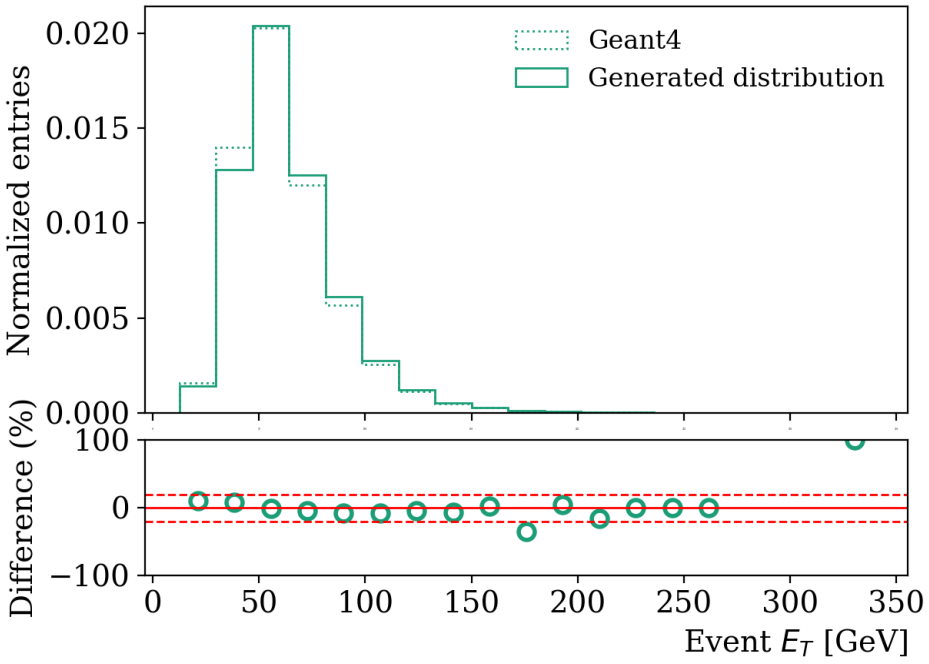


Figure 6.6: Comparison of the sum of hit energies $E_T$ in the ECAL layer. The dashed red lines indicate 20% deviation intervals of the generated samples when compared to GEANT4 predictions.

bution indicates a computed 1-Wasserstein distance of $W_1^{E_T} = 1.22$.

In Figure 6.7, we plot the histogram of EMD values [1] obtained for a collection of 20,000 generated jets. We observe an EMD value of $3.720 \pm 0.159$, with the distribution roughly approximating a Gaussian.

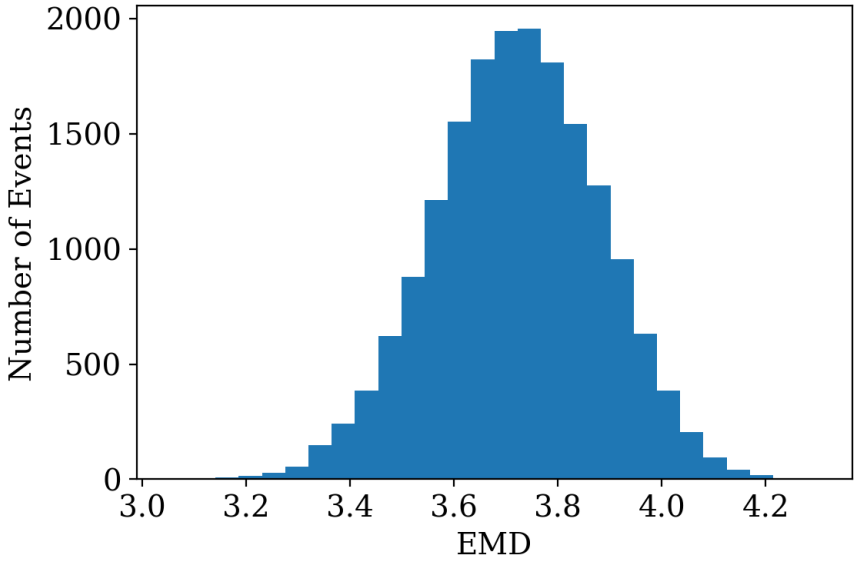


Figure 6.7: A histogram of the Earth Mover's Distance values computed between 20,000 generated and real jets.

We therefore can safely conclude that diffusion models are a promising direction to simulate low-level detector deposits for particle jets. Our model is able to successfully reproduce boosted jets with fairly high fidelity. For example, a visual inspection of the generated samples shows that the model spans the range of real jet distributions accurately in terms of the topology as well as the energy scale. A closer look at the distributions of total event energies reveals that the model is also able to faithfully models the energies seen in actual jets. We also quantify these results by using the 1-Wasserstein distance, and its generalization, the Earth Mover's Distance. Ultimately, these positive results pave the way for further investigations into using diffusion models on low-level detector data. Further work also needs to be done to integrate all three detector channels into a single run of the generation step.

## 6.4 Code Availability

The code written for this thesis can be found at https://github.com/ameya1101/e2e-diffusion/.

[1] We use the following simplified algorithm to compute the EMD between two clouds: https://gist.github.com/ameya1101/7379bba14b112d8380de9698ad679ad3

# Chapter 7

# Conclusions and Future Work

> "I can't give you brains, but I can give you a diploma."
>
> The Wizard of Oz, to the Scarecrow

In this work, we introduce a formulation using score-based generative models for representing, and subsequently simulating, particle jets produced in high energy proton-proton collisions at the Large Hadron Collider. We work with a point cloud representation of particle jets, which tackles the issue of data sparsity in particle detectors by allowing the model to learn directly from particle hits while disregarding any empty cells. The diffusion model learns to reverse a forward process that gradually adds Gaussian noise to pure data over a series of fixed time steps. At each step in the reverse process, we use a powerful Transformer-based backbone network to predict the diffusion velocity for the current time step. Using the transformer architecture, the backbone is network is successfully able to exploit the spatial relationships between different particle hits via an attention mechanism. Once the model is trained, we are able to generate events by applying the reverse process to pure Gaussian noise. Our results indicate that the model is able to accurately learn to reconstruct point cloud events at the Electromagnetic Calorimeter (ECAL) layer. Not only does a visual inspection show that the generated samples are topologically similar to real world events, we find that samples generated from the diffusion model also accurately capture the spatial distribution of energy across hits for the events, as indicated by the distributions plotted for both cases. These results are quantified through evaluation metrics such as the Earth Mover's Distance, or the 1-Wasserstein distance between the hit energies.

Presently, the scope of this work is limited to the ECAL layer. In the future, we recommend an extension of this fast simulation framework to multiple detector channels. One such approach could involve conditioning the diffusion process on a label corresponding to the detector layer we want to generate. For simultaneous generation of multiple layers, we envision constructing event graphs with inter-detector connections and learning to diffuse these hetero-

geneous graphs from pure noise. A well-known limitation of diffusion models is their longer inference time as compared to other iterative and variational generative models. Although the results presented in this work surpass generation times taken by extant tools by many orders of magnitude, they still fall short of the times reported by deep learning-augmented approaches. While distributing the inference task across multiple processing units is certainly an option, we believe a more fundamental architectural change is more significant. We think directions that investigate different backbone networks have the potential to reduce generation time, while maintaining high sample quality. Moreover, a knowledge-distillation framework, as proposed in [64] is a promising improvement to diffusion models that can enable high-fidelity sample generation in as few as eight time steps, as opposed to the 512 steps used in this work.

# Bibliography


[1] GEANT4 collaboration, *GEANT4–a simulation toolkit*, *Nucl. Instrum. Meth. A* **506** (2003) 250.

[2] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol et al., *CaloClouds: Fast Geometry-Independent Highly-Granular Calorimeter Simulation*, 2305.04847.

[3] E. Buhmann, G. Kasieczka and J. Thaler, *EPiC-GAN: Equivariant Point Cloud Generation for Particle Jets*, 2301.08128.

[4] S. Bieringer et al., *Generative Models for Fast Simulation of Electromagnetic and Hadronic Showers in Highly Granular Calorimeters*, *PoS* **ICHEP2022** (2022) 236.

[5] M. Leigh, D. Sengupta, G. Quétant, J.A. Raine, K. Zoch and T. Golling, *PC-JeDi: Diffusion for Particle Cloud Generation in High Energy Physics*, 2303.05376.

[6] V. Mikuni and B. Nachman, *Score-based generative models for calorimeter shower simulation*, *Phys. Rev. D* **106** (2022) 092009 [2206.11898].

[7] ATLAS collaboration, *RECAST framework reinterpretation of an ATLAS Dark Matter Search constraining a model of a dark Higgs boson decaying to two b-quarks*, .

[8] S. Chatrchyan, V. Khachatryan, A.M. Sirunyan, A. Tumasyan, W. Adam, E. Aguilo et al., *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, *Physics Letters B* **716** (2012) 30.

[9] S.N. Bose, *Planck's law and the light quantum hypothesis*, *Journal of Astrophysics and Astronomy* **15** (1994) 3.

[10] E. Fermi, *Zur quantelung des idealen einatomigen gases*, *Zeitschrift für Physik* **36** (1926) 902.

[11] Particle Data Group collaboration, *Review of particle physics*, *Phys. Rev. D* **98** (2018) 030001.

[12] A. Salam and J.C. Ward, *Electromagnetic and weak interactions*, Tech. Rep. IMPERIAL COLL OF SCIENCE AND TECHNOLOGY LONDON (ENGLAND) (1964).

[13] S. Weinberg, *A model of leptons*, *Phys. Rev. Lett.* **19** (1967) 1264.

[14] H. Georgi and S.L. Glashow, *Unified weak and electromagnetic interactions without neutral currents*, *Phys. Rev. Lett.* **28** (1972) 1494.

[15] J. Schwinger, *Quantum electrodynamics. I. a covariant formulation*, *Phys. Rev.* **74** (1948) 1439.

[16] H.D. Politzer, *Reliable perturbative results for strong interactions?*, *Phys. Rev. Lett.* **30** (1973) 1346.

[17] D.J. Gross and F. Wilczek, *Ultraviolet behavior of non-abelian gauge theories*, *Phys. Rev. Lett.* **30** (1973) 1343.

[18] J. Ellis, M.K. Gaillard and D.V. Nanopoulos, *A Historical Profile of the Higgs Boson*, in *The standard theory of particle physics: Essays to celebrate CERN's 60th anniversary*, L. Maiani and L. Rolandi, eds., pp. 255–274 (2016), DOI [1504.07217].

[19] P.W. Higgs, *Broken symmetries and the masses of gauge bosons*, *Phys. Rev. Lett.* **13** (1964) 508.

[20] F. Englert and R. Brout, *Broken symmetry and the mass of gauge vector mesons*, *Phys. Rev. Lett.* **13** (1964) 321.

[21] G.S. Guralnik, C.R. Hagen and T.W.B. Kibble, *Global conservation laws and massless particles*, *Phys. Rev. Lett.* **13** (1964) 585.

[22] Y. Nambu, *Quasi-particles and gauge invariance in the theory of superconductivity*, *Phys. Rev.* **117** (1960) 648.

[23] J. Goldstone, *Field theories with superconductor solutions*, *Il Nuovo Cimento (1955-1965)* **19** (1961) 154.

[24] L. Evans and P. Bryant, eds., *LHC Machine*, *JINST* **3** (2008) S08001.

[25] CMS collaboration, *The CMS Experiment at the CERN LHC*, *JINST* **3** (2008) S08004.

[26] ALICE collaboration, *The ALICE experiment at the CERN LHC*, *JINST* **3** (2008) S08002.

[27] ATLAS collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, *JINST* **3** (2008) S08003.

[28] LHCb collaboration, *The LHCb Detector at the LHC*, *JINST* **3** (2008) S08005.

[29] CMS collaboration, *Interactive Slice of the CMS detector*, .

[30] S. Ehnle, *Uncertainty studies in a measurement of ttH production in the $H \to bb$ channel at CMS*, Ph.D. thesis, 2020.

[31] CMS collaboration, *CMS TriDAS project: Technical Design Report, Volume 1: The Trigger Systems*, Technical design report. CMS.

[32] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer et al., *The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations*, *JHEP* **07** (2014) 079 [1405.0301].

[33] T. Sjöstrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten et al., *An introduction to PYTHIA 8.2*, *Comput. Phys. Commun.* **191** (2015) 159 [1410.3012].

[34] A.M. Turing, *Computing machinery and intelligence*, Springer (2009).

[35] W.S. McCulloch and W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, *The bulletin of mathematical biophysics* **5** (1943) 115.

[36] D.O. Hebb, *The organization of behavior: A neuropsychological theory*, Psychology press (2005).

[37] D.P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, eds., 2015 [1412.6980].

[38] D.P. Kingma and M. Welling, *Auto-encoding variational bayes*, in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, eds., 2014 [1312.6114].

[39] C. Luo, *Understanding diffusion models: A unified perspective*, 2208.11970.

[40] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen and T. Aila, *Analyzing and improving the image quality of stylegan*, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020 [1912.04958].

[41] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair et al., *Generative adversarial nets*, in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Weinberger, eds., vol. 27, Curran Associates, Inc., 2014 [1406.2661].

[42] D.J. Rezende and S. Mohamed, *Variational inference with normalizing flows*, in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, F.R. Bach and D.M. Blei, eds., vol. 37 of *JMLR Workshop and Conference Proceedings*, pp. 1530–1538, JMLR.org, 2015 [1505.05770].

[43] M. Arjovsky, S. Chintala and L. Bottou, *Wasserstein generative adversarial networks*, in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y.W. Teh, eds., vol. 70 of *Proceedings of Machine Learning Research*, pp. 214–223, PMLR, 06–11 Aug, 2017 [1701.07875].

[44] A. Brock, J. Donahue and K. Simonyan, *Large scale GAN training for high fidelity natural image synthesis*, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019 [1809.11096].

[45] S. Zhao, H. Ren, A. Yuan, J. Song, N. Goodman and S. Ermon, *Bias and generalization in deep generative models: An empirical study*, *Advances in Neural Information Processing Systems* **31** (2018) [1811.03259].

[46] J. Ho, A. Jain and P. Abbeel, *Denoising diffusion probabilistic models*, *Advances in Neural Information Processing Systems* **33** (2020) 6840 [2006.11239].

[47] A. Hyvärinen and P. Dayan, *Estimation of non-normalized statistical models by score matching.*, *Journal of Machine Learning Research* **6** (2005) .

[48] Y. Song, S. Garg, J. Shi and S. Ermon, *Sliced score matching: A scalable approach to density and score estimation*, in *Uncertainty in Artificial Intelligence*, pp. 574–584, PMLR, 2020 [1905.07088].

[49] R.T. Chen, Y. Rubanova, J. Bettencourt and D.K. Duvenaud, *Neural ordinary differential equations*, *Advances in neural information processing systems* **31** (2018) [1806.07366].

[50] D. Kingma, T. Salimans, B. Poole and J. Ho, *Variational diffusion models*, *Advances in neural information processing systems* **34** (2021) 21696 [2107.00630v1].

[51] J. Song, C. Meng and S. Ermon, *Denoising diffusion implicit models*, 2010.02502.

[52] T. Salimans and J. Ho, *Progressive distillation for fast sampling of diffusion models*, 2202.00512.

[53] CMS Collaboration, *Tracker-hit-enriched ttjets_hadronicmgdecays_8tev-madgraph*, 2019. 10.7483/OPENDATA.CMS.OPKY.OJMJ.

[54] Greif, Kevin and Lannon, Kevin, *Physics inspired deep neural networks for top quark reconstruction*, *EPJ Web Conf.* **245** (2020) 06029.

[55] M. Cacciari, G.P. Salam and G. Soyez, *The anti-$k_t$ jet clustering algorithm*, *JHEP* **04** (2008) 063 [0802.1189].

[56] M. Andrews, M. Paulini, S. Gleyzer and B. Poczos, *End-to-End Physics Event Classification with CMS Open Data: Applying Image-Based Deep Learning to Detector Data for the Direct Classification of Collision Events at the LHC*, *Comput. Softw. Big Sci.* **4** (2020) 6 [1807.11916].

[57] M. Andrews, J. Alison, S. An, P. Bryant, B. Burkle, S. Gleyzer et al., *End-to-end jet classification of quarks and gluons with the CMS Open Data*, *Nucl. Instrum. Meth. A* **977** (2020) 164304 [1902.08276].

[58] G. Louppe, K. Cho, C. Becot and K. Cranmer, *QCD-Aware Recursive Neural Networks for Jet Physics*, *JHEP* **01** (2019) 057 [1702.00748].

[59] P.T. Komiske, E.M. Metodiev and M.D. Schwartz, *Deep learning in color: towards automated quark/gluon jet discrimination*, *JHEP* **01** (2017) 110 [1612.01551].

[60] A. Aurisano, A. Radovic, D. Rocco, A. Himmel, M.D. Messier, E. Niner et al., *A Convolutional Neural Network Neutrino Event Classifier*, *JINST* **11** (2016) P09001 [1604.01444].

[61] A.Q. Nichol and P. Dhariwal, *Improved denoising diffusion probabilistic models*, in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, eds., vol. 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171, PMLR, 18–24 Jul, 2021 [2102.09672].

[62] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R.R. Salakhutdinov and A.J. Smola, *Deep sets*, in *Advances in Neural Information Processing Systems*, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan et al., eds., vol. 30, Curran Associates, Inc., 2017 [1703.06114].

[63] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez et al., *Attention is all you need*, in *Advances in Neural Information Processing Systems*, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan et al., eds., vol. 30, Curran Associates, Inc., 2017 [1706.03762].

[64] V. Mikuni, B. Nachman and M. Pettee, *Fast Point Cloud Generation with Diffusion Models in High Energy Physics*, 2304.01266.

[65] M. Tancik, P.P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal et al., *Fourier features let networks learn high frequency functions in low dimensional domains*, in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, (Red Hook, NY, USA), Curran Associates Inc., 2020 [2006.10739].

[66] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan et al., *Pytorch: An imperative style, high-performance deep learning library*, in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, eds., vol. 32, Curran Associates, Inc., 2019 [1912.01703].