Politecnico di Torino

Master of Science in Data Science and Engineering

Master Thesis

Lorentz-invariant augmentation for high-energy physics deep learning models

Supervisors
prof. Apiletti Daniele
dott. Monaco Simone

Candidate
Sebastiano Barresi

December 2023

Abstract

In recent years, machine learning models for Jet tagging in high-energy physics have gained considerable attention. However, many existing approaches overlook the physical invariants that jets must adhere to, particularly the fundamental spacetime symmetry governed by Lorentz transformations. Setting this statement as the starting point of this work, it is proposed a model-agnostic training strategy that incorporates theory-guided data augmentation to simulate the effects of Lorentz transformations on jet data.

The study starts with focusing on the state-of-the-art baseline ParticleNet, a neural network architecture designed for the direct processing of particle clouds for Jet tagging. To evaluate the effectiveness of the proposed approach, several experiments are conducted with different augmentation strategies and assess the performance of the augmented models on the widely used top-tagging and quark-gluon reference datasets. The results show that even a small application of the data augmentation strategy increases the robustness of the model to Lorentz boost attacks, i.e., high transformation β. While the accuracy of the baseline model decreases rapidly with increasing intensity of the transformation β, the augmented models exhibit more stable performance. Remarkably, models that underwent a moderate level of augmentation demonstrated a statistically significant performance boost on transformations beyond the ones seen at train time. Then the same experimental setup is applied to a second state-of-the-art baseline LorentzNet, a neural network architecture developed to be invariant to Lorentz transformations by design. The performance of the model are also evaluated both on the top quark tagging and quark-gluon reference datasets, making possible a full comparison between each experimental setup applied to the two chosen models. The results shows that LorentzNet is more robust to Lorentz boost attacks than ParticleNet, as it is expected to be. Nevertheless the application of the data augmentation strategy to an already invariant architecture, tends to further increase the robustness of the model.

This finding highlights the potential of the model-agnostic data augmentation strategy in enhancing model accuracy and robustness while preserving the essential physical properties of the jets.

Contents

Chapter 1

Introduction

This chapter presents the main issues addressed by this master's thesis. The research project focuses on the Theory-Guided Data Science area, which employs techniques to enhance machine learning and deep learning models with domain knowledge. The task at hand is Jet tagging, where the goal is to classify particles produced by collisions in particle accelerators in the field of high-energy particle physics. Specifically, the aim is to impart Lorentz invariance, a physical property of jets, to an unaware model. To this end, it has been selected to use the data augmentation technique, a method yet to be explored in the literature.

1.1 Theory-Guided Data Science

The foundation for this master's thesis lies within the realm of Theory-Guided Data Science [1], a branch emerging from the extensive and highly popular field known as Data Science.

From satellites in space to wearable computing devices and from credit card transactions to electronic health-care records, data has become ubiquitous in everyday life [2]. The capacity to gather, retain, and retrieve vast amounts of information is now a fundamental feature of contemporary society. The exponential growth of information is being propelled by improved sensor technologies, more powerful computing platforms, and greater online connectivity. The analysis of data is becoming increasingly crucial, and this is predicted to continue in the near future. This era in history may be known as the "golden age of data science" as we progress further into the twenty-first century.

In recent times, this field has become increasingly vital in the development of scientific knowledge. Throughout history, the scientific process has advanced by initially creating hypotheses or theories and subsequently gathering data to verify or disprove them. In the era of big data, the continuous collection of extensive information without a particular theory or hypothesis in mind could present further opportunities for uncovering new knowledge. Based on the achievements of data science in domains where Internet-scale data is accessible (with billions or even trillions of samples), for example, natural language translation, optical character recognition, object tracking, and lately, autonomous. With advancements in driving technology, there is an increasing expectation of similar achievements in scientific fields. However, purely data-driven approaches can have their limitations or produce unsatisfactory outcomes in various situations. For instance, an evident case is when there is inadequate data to develop models that perform well and have enough generalization. Another crucial aspect to consider is that a solely data-driven model may not adhere to constraints imposed by natural laws or regulations, which are essential for trustworthy AI.

Representing relationships among physical variables is a prevalent challenge in scientific fields. For instance, such relationships can exist between variables like the combustion pressure and launch velocity of a rocket or the shape of an aircraft wing and its corresponding air drag. The standard practice for illustrating such connections is to adopt theory-based models, grounded in scientific knowledge, which identify cause-effect associations between variables that have either been experimentally proven or deduced from basic principles. These models range from solving closed-form equations (for instance, using the Navier–Stokes equation to study laminar flow) to running computational simulations of dynamical systems (such as the use of numerical models in climate science, hydrology, and turbulence modeling). An alternative approach involves using a set of training examples with input and output variables for learning a data science model that can automatically extract relationships between them. Theory-based and data science models are two opposing approaches to knowledge discovery, each relying solely on one of the two sources of information available in any scientific problem, either scientific knowledge or data. Despite their individual strengths, theory-based and data science models have limitations when applied to scientific problems of great relevance where both theory and data are currently lacking. Some scientific problems involve complex processes that are not yet entirely understood. In such circumstances, theory-based models are frequently compelled to make numerous simplifying suppositions regarding the physical mechanisms, leading to not only inadequate performance but also rendering the model arduous to comprehend and analyze. If "black-box" data science models were to be utilized in scientific research, a distinctive collection of challenges would manifest due to the limited capacity of the data at hand to represent the intricate territory of hypotheses encountered. Moreover, since most data science models only capture associative relationships between variables, they do not fully achieve the objective of understanding the causative relationships in scientific problems. Therefore, in complex scientific applications, knowledge discovery cannot be deemed sufficient with a data-only or a theory-only approach. Instead, it is crucial to examine the continuum between theory-based and data-driven models, wherein both theory and data are utilized in synchronization.

These problems have prompted more research on enhancing machine learning models by incorporating prior knowledge into the learning procedure. Despite the fact that integrating knowledge into machine learning is prevalent, for example, through labeling or feature engineering, a significant gap still needs to be made. As a consequence, there is a rising interest in the incorporation of additional knowledge, particularly in the form of formal knowledge representations. For instance, constraints such as logic rules [3, 4], or algebraic equations [5, 6] have been incorporated into loss functions. Knowledge graphs have the ability to enrich neural networks with relevant information regarding interconnections between instances [7]. This is particularly significant when it comes to image classification [8]. In addition, physical simulations have been employed to improve training data [9]. The recent growth of research activities shows that the combination of data- and knowledge-driven approaches has become relevant in more and more areas. All methods that endeavor to integrate theory-based (e.g., laws of physics) domain expertise into blind, data-driven models can be categorized under the paradigm of Theory-Guided Data Science (TGDS), as depicted by Karpatne et al. [1].

Von Rueden et al. [10] categorized each approach inherent to the TGDS paradigm according to (i) the source of the integrated knowledge, (ii) the representation of the knowledge, and (iii) its integration, i.e., where it is integrated into the learning pipeline.

(i) Knowledge source: The Knowledge source category pertains to the origin of prior knowledge integrated into machine learning. According to the authors, the source of prior knowledge can arise from an established knowledge domain or from individuals with respective experience. The authors also affirm that prior knowledge often stems from the sciences or is a form of world or expert knowledge. Consequently, they propose the below-mentioned categorization.

- *Scientific Knowledge.* Typically, this category of study is formalized and validated through scientific experiments and analytical demonstrations. It encompasses all subjects within science, technology, engineering, and mathematics.

- *World Knowledge.* This type of knowledge is typically intuitive and pertains to human reasoning about the observable world, such as the fact that a feline possesses two ears and can vocalize with a meow. This category also encompasses linguistics, with syntax and semantics being prime examples.
- *Expert Knowledge.* Expert knowledge tends to be exclusive, lacking formalization. Formalization can be achieved through interfaces between humans and machines and validated through the expertise of a specialist group.

(ii) Knowledge representation: The Knowledge representation category corresponds to the formalized element of the prior information. Different representations can be adopted depending on the knowledge available for each specific task. The proposed alternatives are briefly described below.

- *Algebraic Equations.* Algebraic Equations convey knowledge as relationships of equality or inequality between mathematical expressions comprised of variables or constants. They serve to define general functions or to restrict variables to a feasible set and are occasionally referred to as algebraic constraints.
- *Differential Equations.* Differential Equations encompass a subset of algebraic equations that detail the connections between functions and their derivatives in terms of space or time. Two well-known instances are the heat equation, which is a partial differential equation (PDE), and Newton's second law, an ordinary differential equation (ODE). In both situations, there is a set of functions, which may be empty, that solve the differential equation for a given initial or boundary condition.
- *Simulation Results.* Simulation Results depict the numerical output of a computer simulation, which approximates the behavior of a real-world

process. A simulation engine solves a mathematical model using numerical methods and generates outcomes for situation-specific parameters. Its numerical output, the simulation result, serves as the ultimate representation of knowledge. Illustrative instances include the flow field of a simulated fluid or depictions of simulated traffic scenes.

- *Spatial Invariances.* Spatial Invariances refer to properties that remain unchanged during mathematical transformations like translations and rotations. If a geometric object remains unchanged under certain transformations, it exhibits symmetry (such as a triangle that is symmetrical under rotation). A function is capable of. A function can be called invariant, if it has the same result for a symmetric transformation of its argument. Equivariance is a related property that is tied to invariance.
- *Logic Rules.* Logic offers a method for formalizing facts and relationships and enables the translation of everyday language (e.g., `if A THEN B`) into formal logic regulations (e.g., $\mathtt{A} \Rightarrow \mathtt{B}$). Generally, Logic Rules comprise Boolean expressions (`A, B`) linked with logical connectives ($\wedge, \vee, \Rightarrow$). The terms logic constraints or logic sentences may also be used interchangeably.
- *Knowledge Graphs.* A graph is a pair (V, E), where V represents the vertices and E represents the edges connecting them. In a Knowledge Graph, vertices (or nodes) typically describe concepts, while edges depict abstract relationships between these concepts. In a standard weighted graph, edges determine the strength and polarity of the relationship between nodes.
- *Probabilistic Relations.* The central idea of Probabilistic Relationships pertains to a random variable X, from which one can draw samples x based on an underlying probability distribution $P(x)$. Two or more random variables X and Y can exhibit interdependence through their joint distribution $(x, y) \sim P(X, Y)$. Assumptions regarding the conditional can serve as prior knowledge in this scenario. Prior knowledge may include

assumptions regarding the conditional independence or the correlation structure of the random variables or even a comprehensive description of the joint probability distributions.

- *Human Feedback*. Human Feedback refers to technologies that facilitate knowledge transformation through direct interfaces between people and machines. The selection of input methods determines how information gets transmitted. Common input methods include keyboard, mouse, and touchscreen, as well as speech and computer vision, such as motion-tracking devices. In principle, knowledge could also be transferred directly through brain signals using brain-computer interfaces.

(iii) Knowledge integration: The category of Knowledge integration refers to the point in the machine learning pipeline where knowledge is incorporated. The Authors' survey of the literature found that integration methods can be organized into four components: training data, hypothesis set, learning algorithm, and final hypothesis. The following gives a first conceptual overview.

- *Train data*. A common method of integrating knowledge into machine learning is by embedding it in the training data. While traditional machine learning employs the classic approach of feature engineering to create relevant features based on expertise, our definition of an informed approach involves utilizing hybrid information from both the original data set and an external, separate source of prior knowledge. This additional source of prior knowledge facilitates the accumulation of information, thus generating a secondary data set that can be utilized in conjunction with or added to the original training data. A notable technique to achieve this is through simulation-assisted machine learning. Learning where the training data is expanded with simulation results.

- *Hypothesis set.* Incorporating knowledge into the hypothesis set is prevalent, typically accomplished through defining the architecture and hyperparameters of a neural network. For instance, a convolutional neural network utilizes knowledge of object location and translation invariance in images. More generally, knowledge can be integrated through the selection of model structure. One notable example is the development of a network architecture that incorporates a mapping of knowledge elements. as symbols of a logical rule–to specific neurons.
- *Learning algorithm.* Learning algorithms often involve a loss function that can be altered with additional knowledge, such as creating a suitable regularizer. In informed machine learning, algebraic equations, such as the laws of physics, can be integrated by adding loss terms.
- *Final hypothesis.* The final hypothesis of a learning pipeline can undergo benchmarking or validation. For instance, discrepancies between predictions and known constraints may be disregarded or flagged as questionable to ensure that findings align with prior knowledge.

This taxonomy serves as a classification framework for Theory-Guided Data Science and structures approaches according to the three above analysis questions about the Knowledge source, Knowledge representation and Knowledge integration. The Authors combined this taxonomy with a Sankey diagram, reported in Figure 1.1, in which the paths connect the elements across the three dimensions and illustrate the approaches that they found, based on a comparative and iterative analysis of relevant literature. The broader the path, the more papers adopted the approach. Main paths (at least four or more papers with the same approach across all dimensions) are highlighted in darker grey and represent central approaches of Theory-Guided Data Science.

Following the aforementioned taxonomy, this study belongs to the branch of

Scientific Knowledge in terms of knowledge source. As far as knowledge representation is concerned, it can be defined as the incorporation of algebraic equations and spatial invariance. Regarding knowledge integration, it can be classified as an application for training data.

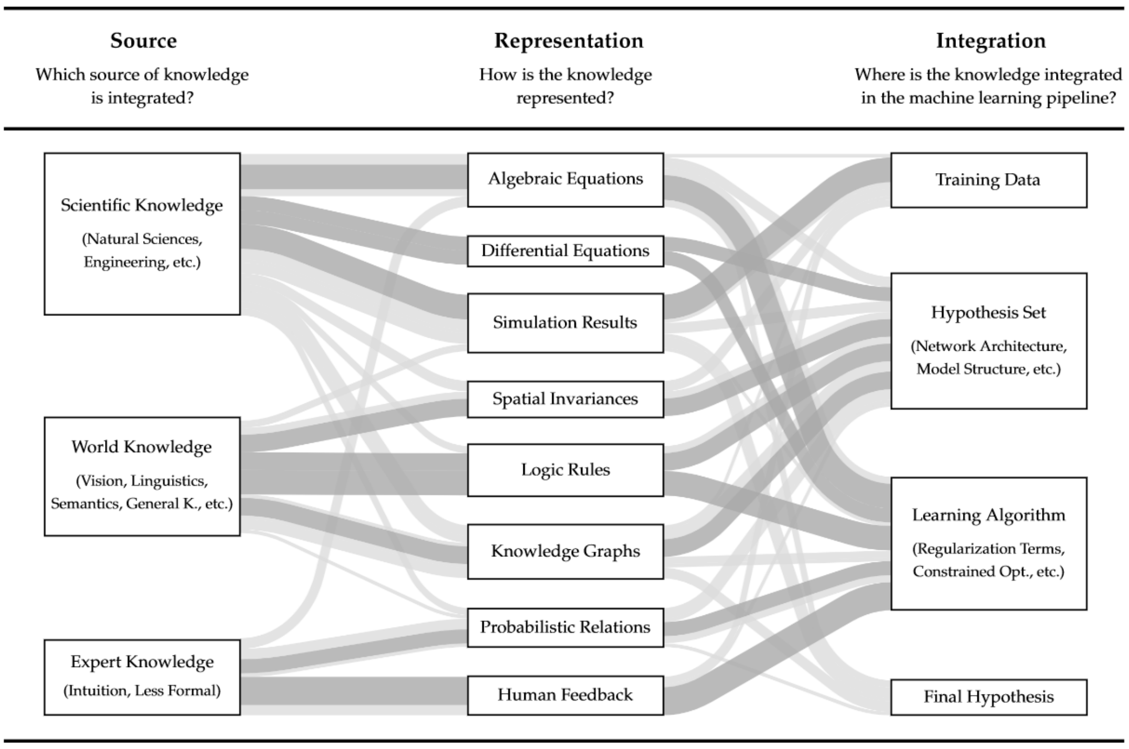

Source
Which source of knowledge is integrated?
Representation
How is the knowledge represented?
Integration
Where is the knowledge integrated in the machine learning pipeline?
Scientific Knowledge
(Natural Sciences, Engineering, etc.)
World Knowledge
(Vision, Linguistics, Semantics, General K., etc.)
Expert Knowledge
(Intuition, Less Formal)
Algebraic Equations
Differential Equations
Simulation Results
Spatial Invariances
Logic Rules
Knowledge Graphs
Probabilistic Relations
Human Feedback
Training Data
Hypothesis Set
(Network Architecture, Model Structure, etc.)
Learning Algorithm
(Regularization Terms, Constrained Opt., etc.)
Final Hypothesis

Figure 1.1: The diagram aims to illustrate the most frequent route within the Theory-Guided Data Science approach. Main paths are highlighted in darker grey. *The Figure is taken from* [10].

1.2 Jet tagging

Within the context of Theory-Guided Data Science, this master's thesis is developed in the field of High Energy Physics (HEP), particularly discussing and developing the Jet tagging task.

The field of high-energy physics is devoted to the study of the elementary constituents of matter. The field strives to unravel the fundamental properties of the physical universe by exploring the structure of matter and the laws that govern its interactions. High-energy physicists rely on modern accelerators, which collide protons and/or antiprotons to generate exotic particles that only occur at extremely high energy densities. Observing these particles and measuring their properties may uncover crucial insights regarding the fundamental nature of matter [11]. These discoveries demand potent statistical methods, with machine learning tools playing an essential role, so High Energy Physics (HEP) is entering a new data-driven era [12].

Many of the applications in the scientific community are connected to the top quark due to its extensive phenomenology both within and beyond the standard model of particle physics (SM). The ATLAS [13] and CMS [14] collaborations feature numerous top quark physics applications that can be divided into three paradigms. Firstly, machine learning (ML) applications are utilized as tagging algorithms to detect top quarks. Secondly, ML is used for the reconstruction of top quark properties. Finally, machine learning is utilized to construct observables that possess a significant separation between signal and background processes, thereby further enhancing the statistical power of analyses [15]. The initial application domain outlined above is thus identified as Jet tagging: the problem of discrimination and identification of high energy jet-like objects observed at the Large Hadron Collider (LHC) [16].

The jets are the experimental signatures of quarks and gluons produced in high-energy processes such as head-on proton-proton collisions. As quarks and gluons have a net color charge and cannot exist freely due to color confinement, they are not directly observed in nature. Instead, they come together to form color-neutral hadrons, a process called hadronization that leads to a collimated spray of hadrons called a jet [17]. Modern techniques often endeavor to simplify representation by exploiting theoretical knowledge or employing deep learning architectures. Nonetheless, current methods frequently neglect significant physical invariants, such as the fundamental spacetime symmetry that inheres in jets. In particular, they do not acknowledge that the physics that underpins jets remains constant even after Lorentz transformations. This last statement forms the basis of this work of master's thesis, which seeks to integrate the physical property of Lorentz invariance into a deep learning model that currently lacks it.

1.3 Lorentz invariance

Lorentz invariance is a fundamental element of modern physics and holds a crucial role in comprehending space, time, and the theory of relativity. The concept is named after the celebrated Dutch physicist Hendrik Lorentz, who first proposed it at the beginning of the 20th century, asserting that the laws of the discipline should remain unaltered under Lorentz transformations. Mathematically, a Lorentz transformation pertains to how the coordinates of an event in spacetime alter when observed from various non-accelerating inertial reference frames. Put simply, the basic rules of physics should appear identical to all observers, regardless of their relative motion. This principle represents a crucial concept in the special theory of relativity, which was formulated by Albert Einstein in 1905 and unifies space and time into a singular four-dimensional continuum known as spacetime.

Lorentz invariance has a significant impact on particle physics, particularly in the progress of quantum field theory (QFT). QFT joins quantum mechanics with special relativity to illustrate the conduct of subatomic particles and their interactions. In QFT, the particles' fields should adhere to the principle of Lorentz invariance, ensuring that their behavior aligns with the core principles of special relativity. This symmetry also imposes restrictions on the kinds of interactions that particles may undergo, determining the makeup of the standard model of particle physics. This framework is effective in depicting the fundamental constituents and forces of nature, except for the force of gravity, which still needs to be reconciled with the quantum description.

In the context of Jet tagging, it is highly desirable that the model be independent of these transformations, much like computer vision architectures should be robust to translations and rotations in input images. By being independent of Lorentz transformations, the model can effectively capture the underlying physical properties of jets regardless of their orientation or boost, leading to improved accuracy and reliability in Jet tagging tasks.

1.4 The contribution

Having set out the framework to which this master's thesis adheres, it is now feasible to explain its composition and the development of this innovative approach.

The task at hand is to enable a deep learning model used for the Jet tagging task to capture the physical property of Lorentz invariance that characterizes particle jets. Therefore, it was decided to employ ParticleNet[18], a model not explicitly designed to have this property, and apply it a novel training strategy involving theory-based transformations of the input data.

The data augmentation technique, explained in chapter 3, involves applying a random Lorentz transformation to the jet data. These transformations are described by a square matrix. They depend on a parameter β, which represents the relative velocity of the jet with respect to the speed of light. The higher the β, the more the original data distribution shifts, leading to a decline in model performance. By integrating basic principles of physics and expert domain knowledge, this approach strives to enhance prediction accuracy whilst upholding Jet tagging precision in a model-agnostic environment.

The encouraging results obtained were subsequently compared to the performance of LorentzNet[19], a deep learning model that is designed to be Lorentz invariant. To enforce the model-agnostic feature of the developed technique, the same training strategy is employed to LorentzNet, which provides valuable insights into the usefulness of a domain-aware augmentation technique on models that are already invariant-aware by design.

The thesis is organized as follows. Chapter 2 provides a comprehensive overview of the related works in the field of jet tagging, highlighting the advancements in various representation approaches and symmetry-preserving deep learning models. In Chapter 3, the proposed methods are presented, focusing on the novel data augmentation technique designed to enhance the performance of state-of-the-art architectures that lack inherent invariance preservation and on the analysis method developed to assess the impact of the aforementioned technique. Chapter 4 provides an overview of the experimental setup. Chapter 5 presents the results of the experiments, evaluating the effectiveness and improvements achieved through the proposed approach. Finally, in Chapter 6, the findings of this study are summarized, discussing the implications, and providing conclusions regarding the potential impact of this work on the field of Jet tagging.

Chapter 2

Related works

This second chapter provides a literature review regarding the Jet tagging task. It explores the theme of deep learning applied to the task, with a following introduction of the most commonly used data representation and architecture in this context.

2.1 Jet tagging with deep learning

Deep learning techniques have been widely proposed to enhance Jet tagging for particle physics experiments [20]. The efficiency and effectiveness of machine or deep learning techniques on jet physics depends heavily on the jet's representation [18]. Therefore, the models and techniques used in this area of research are introduced by subdividing them through the type of representation employed. What follows is a review of the most common representations of jets with an introduction to recently formulated particle cloud representation. Furthermore, new approaches to the Jet tagging problem will be described, including an additional introduction of domain knowledge.

Image-based representation

The origin of image representation lies in the reconstruction of jets using calorimeters, which measure the energy deposition of a jet on finely-grained spatial cells. Treating the energy deposition on each cell as pixel intensity naturally creates an image of the jet. When forming jets from particles reconstructed with full detector information (such as through a particle-flow algorithm), a jet image can be created by mapping each particle onto its corresponding calorimeter cell and summing up the energy when multiple particles are mapped to the same cell. Various studies [21, 22] investigated convolutional neural networks (CNNs) with different architectures and found that they significantly improve performance compared to traditional multivariate methods using observables based on QCD theory. However, the CNN architectures investigated in these studies are generally much shallower compared to state-of-the-art ones used in image classification tasks. Examples of such architectures include ResNet [23] or Inception [24]. Therefore, it remains to be seen if deeper architectures can enhance performance to a greater extent. Although the image-based representation shows promise, it has two main shortcomings. When a jet is measured only by the calorimeter, the image-based representation can include all information without loss. However, once the jet constituent particles are reconstructed, it is unclear how to incorporate additional information about the particles, as it involves combining non-additive quantities (such as particle type) of multiple particles entering the same cell. Moreover, representing jets as images results in a sparse representation. A typical jet contains $O(10)$ to $O(100)$ particles, whereas a jet image needs $O(1000)$ pixels (e.g., 32×32) to capture the jet information fully. As a result, more than 90% of the pixels are blank, making CNNs highly computationally inefficient for analyzing jet images.

Particle-based representation

A more natural approach to representing a jet during particle reconstruction is to view it as a collection of its constituent particles. This strategy permits the inclusion of any feature for each particle, thus rendering it considerably more flexible than the image representation. Although this method is more concise in comparison to the image representation, it is variable in length as each jet may have a different number of particles. A collection of particles is a broad notion. To apply a deep learning algorithm, however, a specific data structure must be selected. Typically, a sequence is preferred, where particles are ordered according to a particular criterion (e.g., decreasing transverse momentum) and assembled in a one-dimensional list. Particle sequences have been used as inputs to address Jet tagging tasks through the utilization of recurrent neural networks (RNNs) [25], 1D CNNs [26], and physics-focused neural networks [27]. One important aspect to consider regarding the sequence representation is that the particles must be arranged in some manner, as the order of the particles is utilized implicitly in the corresponding RNN or 1D CNNs. However, it should be noted that the constituent particles within a jet lack intrinsic order. Therefore, any manually imposed order may prove sub-optimal and could ultimately hinder performance.

Particle cloud representation

An even more natural representation than particle sequences would be an unordered, permutation-invariant set of particles. This method shares all the advantages of particle-based representations, including the flexibility to include any arbitrary features for each particle. This representation is referred to as "particle cloud", which is analogous to the point cloud representation of 3D shapes used in computer vision. They are fundamentally similar since both are essentially unordered sets of entities irregularly distributed in space. In both clouds, the elements are not unrelated individuals but instead are correlated since they represent higher-level objects (i.e., jets or 3D shapes) with rich internal structures. Consequently, deep learning algorithms created for point clouds will probably be advantageous for particle clouds, specifically jets. The Deep Sets [28] and Dynamic Graph CNN [29] architectures are then adapted for Jet tagging, resulting in the creation of the Energy Flow Network [30] and the state-of-the-art ParticleNet [18]. The Energy Flow Network relies on the Deep Sets approach, which has the limitation of not explicitly exploit the local spatial structure of particle clouds, processing only the particle clouds in a global way. ParticleNet is derived from the Dynamic Graph CNN architecture, which overcomes this limitation introducing the process of particle-based features. Nevertheless, the architecture lacks of the preservation of jets physical properties as the Lorentz invariance, which will be the focus of this master's thesis work.

Integrating domain knowledge

Recently, researchers have increasingly focused on integrating inductive biases based on physics principles into architectural design. This includes implementing the Lund jet plane [31], the Lorentz group symmetry [32, 19], and rotational symmetry, among others. LorentzNet is the state-of-the-art for symmetry-preserving deep learning models for jet tagging. Relying on the particle cloud representation, achieves the best tagging performance and improves over existing state-of-the-art algorithms such as ParticleNet. The performance of the model comes at the cost of an increasing demand for resources, training and inference time.

Deep Learning in the search for new physics

Deep learning-based Jet tagging algorithms have been widely adopted in real-world data analysis at the LHC. Using ParticleNet, CMS achieves the first observation of Z boson decay to a pair of charm quarks at a hadron collider. The network is also used by CMS to probe the quartic interaction between the Higgs and vector bosons, indirectly confirming its existence for the first time. Clearly, advances in Jet tagging have a crucial impact on accelerating the comprehension of elementary particles, the fundamental components of nature. In this framework, the aim of this study is to introduce the physical principle of Lorentz invariance into a model such as ParticleNet, which currently lacks this feature, by using a novel training technique through data augmentation. The purpose of this approach is to align the performance of the model with others that possess this property by architectural design.

Chapter 3

Methods

The third chapter presents the methods utilized to achieve the objective of the master's thesis. It initiates with a theoretical presentation of the Lorentz group and its associated invariance property. Graph neural networks are subsequently introduced, outlining their key characteristics and application in Jet tagging. Then the architectures of the models chosen to conduct this study are introduced. To conclude the data augmentation technique employed is meticulously analyzed, followed by an in-depth discussion of the evaluation method used to measure its effectiveness.

3.1 The Lorentz group

The mathematical formulation of the principles of special relativity that govern the behavior of jet particles involves unifying space and time in a 4-dimensional spacetime framework. In this framework, the Minkowski metric replaces the classical Euclidean dot product. Within this paradigm, the Lorentz group represents the transformations of space and time coordinates between different inertial reference frames, with the key requirement that the underlying physics remains unchanged in all inertial frames.

The Minkowski metric

Considering the 4-dimensional space-time $\mathbb{R}^4$ with basis $e_{i_{i=0}}^{3}$, it is defined a bilinear form η $\mathbb{R}^4$Ö$\mathbb{R}^4 \to \mathbb{R}$ as follows. For $u, v \in \mathbb{R}^4$ it is setted $\eta(u,v) = u^T J v$, where $J = diag(1, -1, -1, -1)$ is the Minkowski metric. The Minkowski inner product of two vectors $u = (t, x, y, z)$ and $v = (t', x', y', z')$ is defined as $\langle u, v\rangle = \eta(u, v) = tt' - xx' - yy' - zz'$. The Minkowski norm of a vector $u = (t, x, y, z)$ is defined to be $\|u\| = \sqrt{\eta(u,u)} = \sqrt{t^2 - x^2 - y^2 - z^2}$.

Lorentz transformation

Lorentz transformations are linear transformations Δ^{μ}_{ν} that preserve the bilinear form η. Restricting the inertial frames to be positively oriented and positively time-oriented, the orthochronous Lorentz group is obtained, denoted as $SO(1{,}3)^+$. The infinitesimal transformations in $SO(1{,}3)^+$ include six degrees of freedom. From the physics interpretation, these include three types of rotation in the space dimensions and three types of Lorentz boosts involving the time dimension (denoted as *x-t, y-t* and *z-t* boost).

Given two inertial frames, the relative velocity $\beta = v/c$, where c is the speed of light, and the boost factor $\gamma = (1 - \beta^2)^{-1/2}$, taking *x-t* Lorentz boost and *x-y* rotation as examples, then the *x-t* boost is represented by the matrix

$$Q_\beta = \begin{pmatrix} \gamma & -\gamma\beta & 0 & 0 \\ -\gamma\beta & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.1}$$

and the *x-y* rotation has the form

$$Q_\alpha = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.2}$$

The above-discussed matrix formalization of the Lorentz transformation constitutes the key element enabling the development of the proposed data-augmentation strategy.

3.2 Graph Neural Network for particle cloud

Graphs are widely used in science, engineering, and various problem domains [33]. A graph $G = (V, E)$ is essentially comprised of a group of nodes V and edges E that are adjacent to pairs of nodes. Richer types of graphs include several special cases. For instance, one is the tree, where only one sequence of edges joins any two nodes. Another is the directed graph, where the two nodes associated with an edge are ordered. Additional ones are attributed graphs, which include node-level, edge-level, or graph-level attributes, multigraphs, where multiple edges may exist between nodes, and hypergraphs, where more than two nodes are associated with an edge. Crucially, graphs provide a natural and potent method for expressing numerous intricate systems, such as trees that depict the evolution of species or the hierarchical makeup of sentences. They also include lattices and meshes to exhibit the regular and irregular discretization of space, respectively, and dynamic networks to relay traffic on roads and social relationships over time. A jet may be represented as a graph when its constituent particles are viewed as nodes. For the particle with index i, the coordinate of node i in Minkowski space is its 4-momenta vector $v_i = (E^i, p_x^i, p_y^i, p_z^i)$. The scalars, including mass, charge, and particle identity information, that make up the node attributes are denoted by $s_i = (s_1^i, s_2^i, \ldots, s_\alpha^i)$. The essential features for tagging are then contained in $f_i = v_i \oplus s_i$. The edges denote the message passing between two particles, indicating the interaction of two individual sets of particle-wise features. If there is no interaction, there will be no edge between the corresponding nodes. The graph is considered to be fully connected as there are no assumed prior interactions among the particles.

Graph Neural Networks (GNNs), as described in literature [34, 35], are deep learning architectures that implement powerful relational inductive biases for learning functions that operate on graphs. These models use a parameterized message-passing mechanism, propagating information across the graph. This enables the computation of advanced edge, node, and graph-level outputs. Within a GNN, there are typically one or more standard building blocks for neural networks known as fully connected layers. These building blocks are responsible for executing message computations and propagation functions.

Given a graph $G = (V, E)$, assuming L steps in total, the l-th message passing step on the graph can be described as [36]:

$$m_i^{l+1} = \sum_{j \in [\mathbb{N}(i)]} M_l(h_i^l, h_j^l, e_{ij}) \tag{3.3}$$

$$h_i^{l+1} = U_l(h_i^l, m_i^{l+1}) \tag{3.4}$$

where $h_i^0 = f_i$ is the input feature, e_{ij} is the edge feature, $\mathbb{N}(i)$ is the set of neighbors of node i, and M_l, U_l are neural networks. For a classification problem, the output $\hat{y}$ can be obtained by applying the softmax function after decoding $h_i^L; i \in [N]$.

3.2.1 ParticleNet

ParticleNet [18] is a customized neural network architecture that operates directly on particle clouds for Jet tagging. Motivated by the success of convolutional neural networks (CNNs), the Authors implemented a comparable strategy for learning from point cloud data. However, standard convolution operations cannot be utilized for point clouds due to the non-uniform distribution of points, unlike the uniform grid points in an image. As a result, the foundation for convolution, that is the "local patch" of each point upon which the convolution kernel functions, must be established for point clouds. Additionally, a conventional convolution operation, expressed as $\sum_j K_j x_j$ where K represents the kernel and x_j denotes the features of each point, is not invariant when the points are permuted. Consequently, the convolution form should be adjusted to maintain the permutation symmetry of the point clouds.

Network architecture

The Edge convolution. Recently, Ref. [29] proposed the edge convolution ("EdgeConv") as a convolution-like operation for point clouds. EdgeConv represents a point cloud as a graph, where the points themselves are the vertices and the edges are connections between each point and its k nearest neighboring points. This creates a local patch necessary for convolution for each point, defined as the k nearest neighboring points connected to it. The EdgeConv operation for each point x_i then has the form

$$x_i^{'} = \square_{j=1}^{k} h_{\Theta}(x_i, x_{i_j}) \tag{3.5}$$

where $x_i \in \mathbb{R}^F$ denotes the feature vector of the point x_i and $i_1, \ldots, i_k$ are the indices of the k nearest neighboring points of the x_i. The edge function h_Θ is some function parametrized by a set of learnable parameters Θ, and $\square$ is a channel-wise symmetric aggregation operator, e.g., max, sum, or mean. The parameters Θ of the edge function are shared for all points in the point cloud. This, together with the choice of a symmetric aggregation operator $\square$, makes EdgeConv a permutationally symmetric operation on point clouds. The ParticleNet Authors have used a specialized form of the edge function following the work of [29],

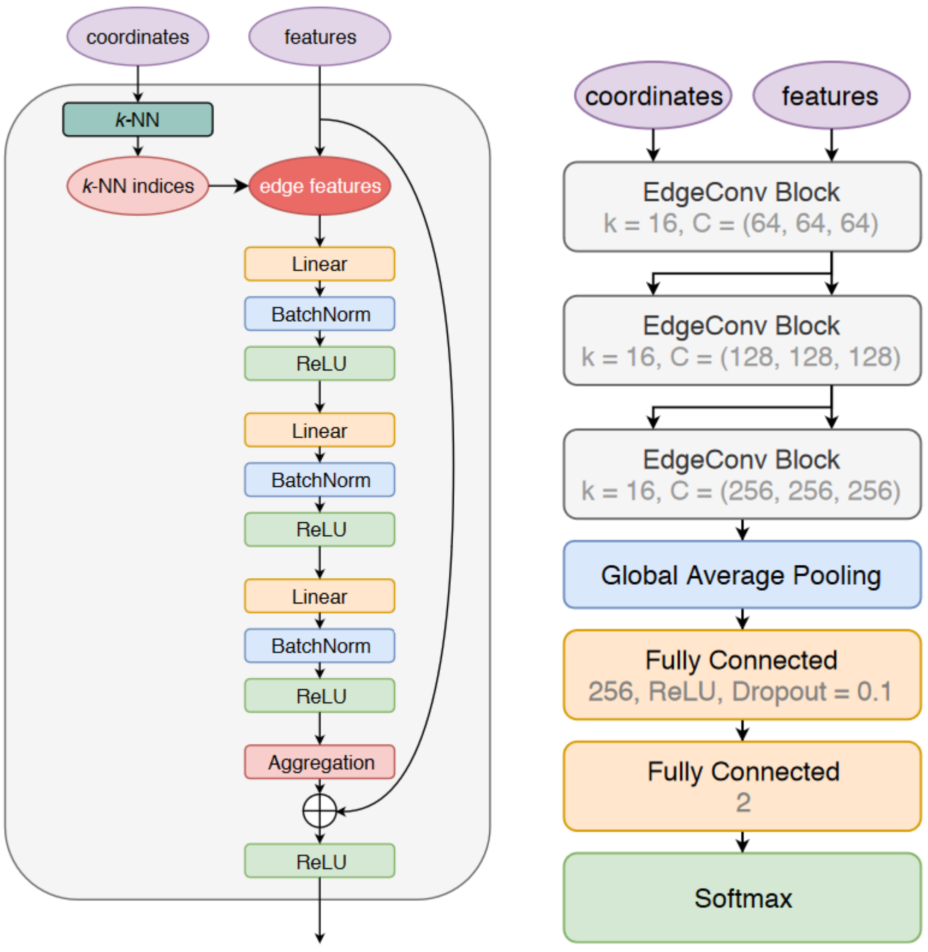
$$h_{\Theta}(x_i, x_{i_j}) = h_{\Theta}^{'}(x_i, x_{i_j} - x_i) \tag{3.6}$$

where the feature vectors of the neighbors, x_{i_j}, are substituted by their differences from the central point x_i and $h_{\Theta}^{'}$ can be implemented as a multilayer perceptron (MLP) whose parameters are shared among all edges. For the aggregation operator $\square$, the choice is set to be the mean, as it showed better performance than the max operation used in the original paper.

A critical aspect of the EdgeConv operation is its ease of stacking, similar to regular convolutions. This is because EdgeConv can be interpreted as a point cloud mapping to another point cloud with an equal number of points, potentially altering the feature vector dimension for each point. Subsequently, an additional EdgeConv operation can be applied, facilitating deep network construction using EdgeConv operations to learn point cloud attributes hierarchically. The stackability of EdgeConv operations presents another possibility. Essentially, the feature vectors that EdgeConv learns can serve as new coordinates for the points in a latent space. Consequently, the distances between points, which are used to determine the k-nearest neighbors, can then be calculated in this latent space. In other words, EdgeConv operations can dynamically teach the proximity of points. Ref. [29] demonstrates that this leads to better performance than keeping the graph static.
The model. The ParticleNet architecture extensively utilizes EdgeConv operations while also adopting the dynamic graph update approach. Several design choices were made in ParticleNet as compared to the original DGCNN to improve the suitability for the Jet tagging task. These design choices include alterations in the number of neighbors, configuration of the MLP in EdgeConv, and use of shortcut connections. Figure 3.1 (a) illustrates the structure of the EdgeConv block implemented by the Authors. The EdgeConv block begins by identifying the k closest neighboring particles for each particle, utilizing the "coordinates" input of the EdgeConv block to determine distances. Next, the "edge features" inputs to the EdgeConv operation are constructed from the "features" input utilizing the k nearest neighboring particle indices. The EdgeConv operation is implemented as a three-layer MLP. Each layer comprises a linear transformation, followed by batch normalization, and then a rectified linear unit (ReLU). Inspired by ResNet, each block includes a shortcut connection running parallel to the EdgeConv operation. This allows the input features to pass directly through. An EdgeConv block is defined by two hyperparameters: the number of neighbors (k) and the

number of channels ($C = C_1, C_2, C_3$), which correspond to the units in each linear transformation layer.

The architecture presented by the Authors is illustrated in Figure 3.1 (b), consisting of three EdgeConv blocks. The initial EdgeConv block calculates distances using the spatial coordinates of the particles in the pseudorapidity-azimuth space, while the following blocks use learned feature vectors as coordinates. The number of nearest neighbors k remains fixed at 16 across all three blocks, with the number of channels C varying for each EdgeConv block: (64, 64, 64), (128, 128, 128), and (256, 256, 256), respectively. After applying the EdgeConv blocks, a channel-wide global average pooling operation aggregates the features learned over all particles within the cloud. Next, a fully connected layer with 256 units and the ReLU activation function is utilized. To prevent overfitting, a dropout layer with a drop probability of 0.1 is incorporated. A fully connected layer with two units, followed by a softmax function, generates the binary classification output.



(a) (b)

Figure 3.1: (a): The EdgeConv block structure;
(b): The architecture of ParticleNet.
The figures are taken from [18].

3.2.2 LorentzNet

LorentzNet [19] is a symmetry-preserving deep learning model for Jet tagging. Using the particle cloud representation of jets, the model directly scalarizes the input 4-vectors to ensure Lorentz symmetry. Specifically, the authors designed Minkowski dot product attention to aggregate the four 4-vectors with weights learned from Lorentz-invariant geometric quantities under the Minkowski metric. The construction of LorentzNet is guided by the universal approximation theory on Lorentz equivariant mapping, which guarantees LorentzNet's equivariance and universality. Compared to LGN [32], a Lorentz equivariant neural network architecture built with tensor products of Lorentz group representations, which necessitates computationally expensive tensor products of the geometric quantities to achieve notable expressiveness, LorentzNet solely demands the Minkowski inner product of two vectors. Thus, it is more efficient in terms of both training and inference.

Network architecture

The Lorentz Group Equivariant Block. Defining the notation first, the Authors use $h^l = (h_1^l, h_2^l, \ldots, h_N^l)$ to denote the node embedding scalars, and $x^l = (x_1^l, x_2^l, \ldots, x_N^l)$ to denote the coordinate embedding vectors in the l-th LGEB layer. When $l = 0$, x_i^0 equals the input of 4-momenta, and h_i^0 equals the embedded input of the scalar variables s_i. LGEB aims to learn deeper embeddings h^{l+1}, x^{l+1} via current h^l, x^l.

Motivated by the principle of Lorentz invariance, LorentzNet's message passing is expressed as follows. Denoting as m_{ij} the edge message between particle i and j, i.e.,

$$m_{ij}^l = \phi_e(h_i^l, h_j^l, \psi(\|x_i^l - x_j^l\|^2), \psi(\langle x_i^l, x_j^l \rangle)) \tag{3.7}$$

where $\phi_e(\cdot)$ is a neural network and $\psi(\cdot) = sgn(\cdot)log(|\cdot| + 1)$ is to normalize large numbers from broad distributions. The operation $\langle x_i^l, x_j^l \rangle$ is defined as Minkowski dot product, while the $\|x_i^l - x_j^l\|^2$ is included due to the fact that interaction between particles relies on this term.

The Authors further designed a *Minkowski dot product attention* as

$$x_i^{l+1} = x_i^l + c \sum_{j\in[N]} \phi_x(m_{ij}^l \cdot x_j^l) \tag{3.8}$$

where $\phi_x(\cdot) \in \mathbb{R}$ is a scalar function modeled by neural network. The hyperparameter c is introduced to control the scale of x_i^{l+1}.

The scalar features for particle i are forward as

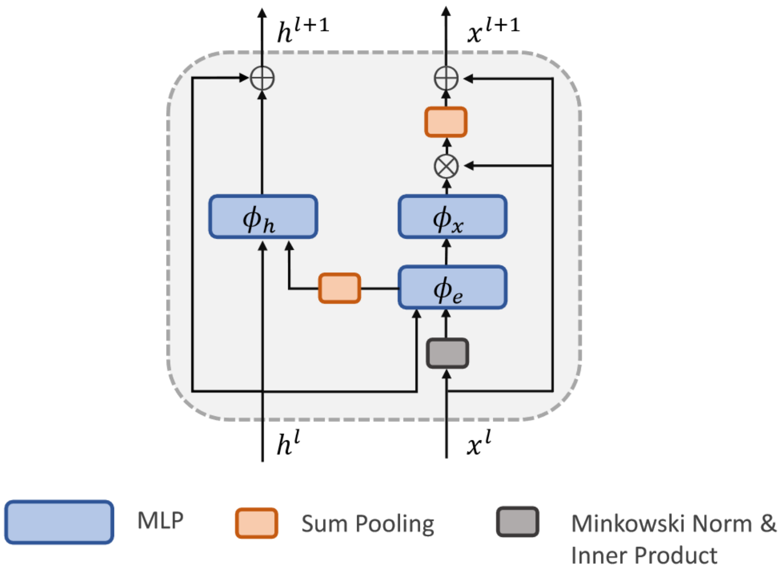
$$h_i^{l+1} = h_i^l + \phi_h(h_i^l, \sum_{j\in[N]} w_{ij} m_{ij}^l) \tag{3.9}$$

where $\phi_h(\cdot)$ is also modeled by neural networks whose output dimension equals the dimension of h_i^{l+1}. For efficient computation, the summation $\sum_{j\in[N]} w_{ij} m_{ij}^l$ is operated to aggregate m_{ij}^l. The neural network $\phi_m(\cdot)$ is introduced to learn the edge significance from node j to node i, i.e., $w_{ij} = \phi_m(m_{ij}^l) \in [0,1]$. This both ensure the permutation invariance and also ease the implementation for jets with a different number of particles.

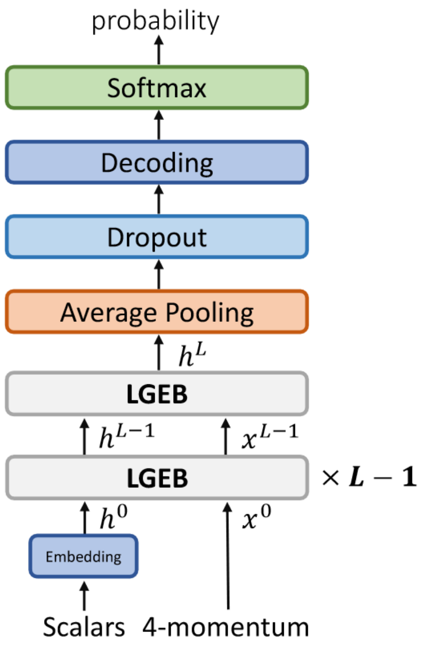
The model. LorentzNet is mainly constructed by the stack of Lorentz Group Equivariant Block (LGEB) along with encoder and decoder layers. The inputs into the network are 4-momenta of particles from a collision event and may include scalars associated with them (such as label, charge, . . .). After stacks of LGEB for L layers, the obtained node embedding h^L is decoded. First, average pooling is used to get

$$h^{av} = \frac{1}{N} \sum_{i\in[N]} h_i^L \tag{3.10}$$

Then, dropout is applied to h^{av} to prevent overfitting. A decoding block with two fully connected layers, followed by a softmax function, is used to generate the output for the binary classification task.



(a)



(b)

Figure 3.2: (a): The Lorentz Group Equivariant Block (LGEB); (b): The architecture of LorentzNet. *The figures are taken from* [19].

3.3 The augmetation strategy

In the discussed point cloud representation of jets, all particles are nodes represented by a set of features, including their 4-momenta vector $v_i = (E^i, p_x^i, p_y^i, p_z^i)$ and other possible additional features such as charge and mass of the particle denoted as $s_i = (s_1^i, s_2^i, \ldots, s_\alpha^i)$. The concatenation $f_i = v_i \oplus s_i$ represents the complete set of features per node.

The data augmentation technique developed to preserve invariances aims to first perform a rotation in three-dimensional space while preserving the direction of the particle beam, and then it applies an *x*-*t* Lorentz boost to a jet. Then, at a particular boost value β and rotational angle α, all jet particles undergo the following transformation:

$$\begin{aligned} v_i' &= Q_\alpha \times v_i \\ v_i'' &= Q_\beta \times v_i' \\ f_i &= v_i'' \oplus s_i \end{aligned} \tag{3.11}$$

where Q_β and Q_α are described in equations 3.1 and 3.2. The degree of preservation of Lorentz invariance can be adjusted by controlling the parameter β_{max}, representing the maximum boost that can be applied to the entire jet. Consequently, β can assume value in the range $[0, \beta_{max})$. To introduce variability, the transformation is applied with a probability p.

Additionally, regardless of weather the boost is performed, the input features can be encoded to match the required features of the model. The transformation is outlined in Algorithm 1.

3.4 The invariance analysis

To evaluate the effectiveness of the proposed method, following the work done in [19], the model's performance are further analyzed through a transformed test set.

Specifically, an x-t Lorentz boost is applied to a sampled test set with different scales of β. As β increases, the divergence between the distributions of the training data that is not transformed and the test data becomes more considerable. Therefore, the task of classification is expected to be more arduous with the increasing of β, because the model has to capture invariants over a wider spectrum.

The analysis produces a visualization of the accuracy of the model in tagging the transformed test samples, for β ranging in [0,1]. This evaluation technique facilitates comparison between the augmented and baseline models, as well as the confrontation between different augmentation strategies. The invariance analysis algorithm is outlined in Algorithm 2. The integration of the aforementioned method and of the presented data augmentation technique into the training pipeline are represented in Figure 3.3.

Algorithm 1: $lorentz\_augmentation(jet, \beta_{\max}, p)$

```
if $random(0,1) < p$ then
    $\beta \leftarrow random(0,1) \cdot \beta_{\max}$;
    $\alpha \leftarrow random(0,180)$;
    $Q_\alpha \leftarrow rotation\_matrix(\alpha)$;
    $Q_\beta \leftarrow lorenz\_transformation(\beta)$;
    $rotated\_jet \leftarrow Q_\alpha \times jet.v$;
    $transformed\_jet \leftarrow concat(Q_\beta \times rotated\_jet \oplus jet.s)$;
end
$jet \leftarrow feature\_transformation(transformed\_jet)$;
$return\ jet$;
```

Algorithm 2: $invariance\_analysis$

```
$betas \leftarrow range(0,1,0.02)$;
for $\beta$ $in\ betas$ do
    $transformed\_test \leftarrow lorentz\_augmentation(test\_set, \beta, p = 1)$;
    $accuracy \leftarrow model(transfomed\_set)$;
    $invariance\_table \leftarrow (\beta, accuracy)$;
end
$return\ invariance\_table$;
```

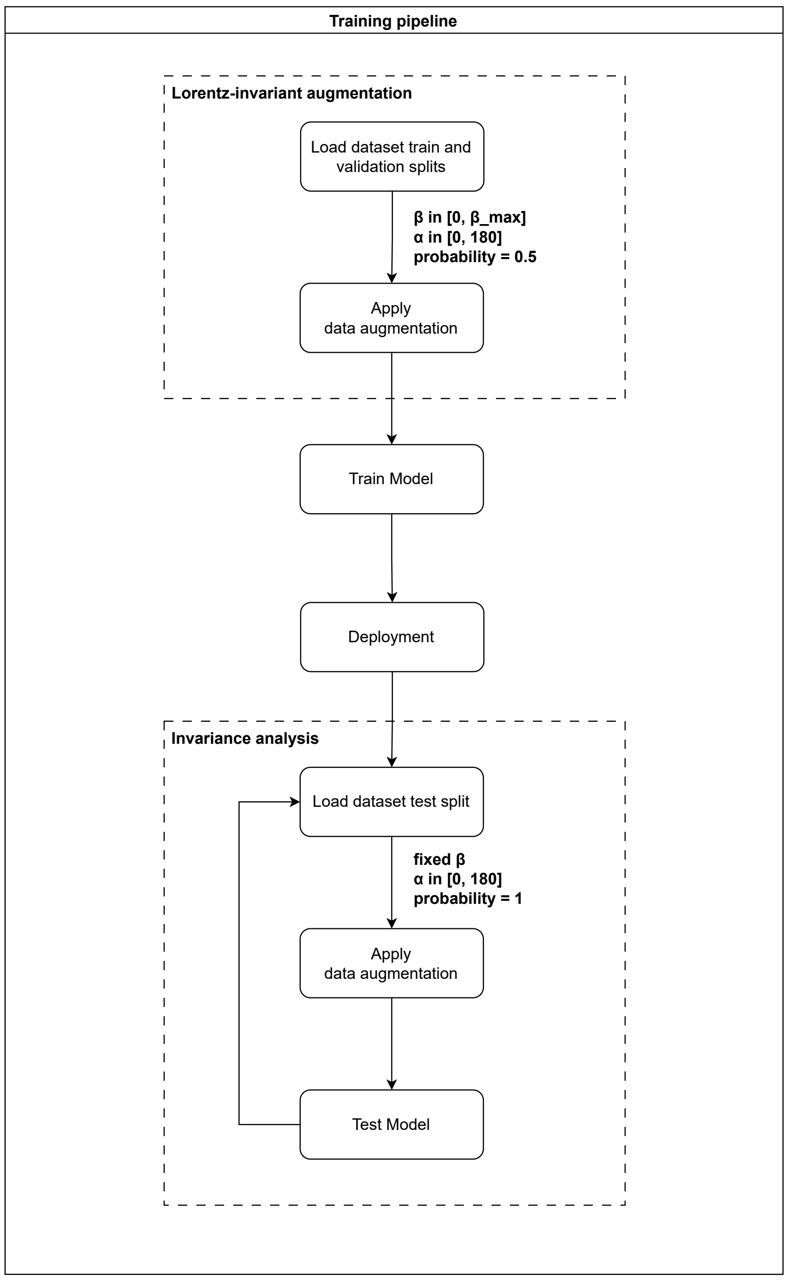


Figure 3.3: The figure presents the training pipeline created for this master's thesis work.

Chapter 4

Experimental setup

Chapter 4 is devoted to outlining the experimental procedures employed in this research. First, there is a description of the datasets utilized to train the two models introduced in Chapter 3. Following this, the training framework is presented, including a discussion of the technological solutions used and the model implementations.

4.1 Datasets

In this study, two benchmark data sets were used to identify jets. These are the Top quark tagging dataset and the Quark-gluon tagging dataset, which were previously used by the Authors of the models selected for this work. This decision allowed for easier implementation and execution of the experiments.

4.1.1 Top quark tagging dataset

The Top quark tagging dataset [37], proposed in [20], contains 1.2M training entries, 400k validation entries, and 400k testing entries, each of these representing a single jet whose origin is either an energetic top quark, a light quark or a gluon.

Consequently, the classification task is a binary classification task. Jets in this dataset are generated using the `Pythia8`[38] Monte Carlo event generator, while the ATLAS detector response is modelled with the `DELPHES`[39] software package. The jets in the reference dataset are clustered using the anti-k_T algorithm, with a radius of $R = 0.8$. For each jet, the 4-momenta are saved in Cartesian coordinates (E, p_x, p_y, p_z) for up to 200 constituent particles selected by the highest transverse momentum. For jets with less than 200 constituents a zero padding is applied.

4.1.2 Quark-gluon tagging dataset

The Quark-gluon tagging dataset [40], proposed in [30], address the task of discriminating jets initiated by quarks and by gluons. The classification task consists in a binary classification task. The signal (quark) and background (gluon) jets are generated with `Pythia8` using the $Z(\to \nu\nu) + (u, d, s)$ and $Z(\to \nu\nu) + g$ processes, respectively. No detector simulation is performed. The final state non-neutrino particles are clustered using the anti-k_T algorithm with $R = 0.4$. The dataset consists of 2 million jets in total, with half signal and half background. The recommended split is followed, obtaining 1.6M training entries, 200k validation entries and 200k testing entries. The major difference is that the Quark-gluon tagging dataset includes the 4-momenta vectors alongside with the particle identification information (PIDs), namely the type of each particle (i.e., electron, photon, pion, ...).

	Top quark tagging	**Quark-gluon tagging**
Number of entries	1.2 million	2 million
Features	4-momenta	4-momenta and PIDs
Task	Binary classification	Binary classification

Table 4.1: The table provides a summary of the datasets insights.

4.2 Training setup

The training setup has been developed taking advantage of the PyTorch Lightning framework [41]. PyTorch Lightning is a deep learning framework designed for professional AI researchers and machine learning engineers seeking maximum flexibility without compromising performance at scale.

Within this framework, the Weight & Biases [42] platform has been implemented and extensively utilized. This platform enables real-time tracking of experiments with live visualizations of metrics, logs, and system statistics. It also includes a tool that allows automatic hyperparameter search, known as "Sweep", which was used in this study to automatically perform the training of various augmentation setups on the same model.

The trainings are carried out on a machine with an Intel Core i9-10980XE CPU and an Nvidia RTX A6000 graphics card.

4.2.1 ParticleNet implementation

The implementation of the ParticleNet model is obtained following the ones proposed in [43] that makes use of the PyTorch Geometric framework [44]. PyTorch Geometric is a library built upon PyTorch to easily write and train Graph Neural Networks, it consists of various methods for deep learning on graphs and other irregular structures.

The model parameters are set according to the authors' original configuration. The EdgeConv blocks in the architecture consist of three layers, with the number of channels set as (64, 64, 64), (128, 128, 128), and (256, 256, 256), respectively. All three blocks have a fixed number of nearest neighbours $k = 16$ to build the graphs over the particle point clouds. After applying the EdgeConv blocks, a channel-wide global average pooling operation aggregates the features learned over all particles within the cloud. Next, a fully connected layer with 256 units and the ReLU activation function is utilized. To prevent overfitting, a dropout layer with a drop probability of 0.1 is incorporated. A fully connected layer with two units, followed by a softmax function, generates the binary classification output.

The input required by the model consists in seven variables derived from the 4-momenta vector, to which are concatenated the PIDs information in the case of the Quark-Gluon tagging task. The preliminary features utilized to compute the inputs to the model are

$$\begin{aligned} p_T &= \sqrt{p_x^2 + p_y^2} \\ \eta &= \arctan \frac{p_z}{E} \\ \phi &= \arctan \frac{p_y}{p_x} \\ \Delta R &= \sqrt{\Delta\eta^2 + \Delta\phi^2} \end{aligned} \tag{4.1}$$

The inputs are listed in Table 4.2.

Variable	Definition	TOP	QG
$\Delta\eta$	difference in pseudorapidity between the particle and the jet axis	✓	✓
$\Delta\phi$	difference in azimuthal angle between the particle and the jet axis	✓	✓
$\log p_T$	logarithm of the particle's transverse momentum	✓	✓
$\log E$	logarithm of the particle's energy	✓	✓
$\log \frac{p_T}{p_T(jet)}$	logarithm of the particle's p_T relative to the jet p_T	✓	✓
$\log \frac{E}{E(jet)}$	logarithm of the particle's energy relative to the jet energy	✓	✓
ΔR	angular separation between the particle and the jet axis	✓	✓
q	electric charge of the particle		✓
isElectron	if the particle is an electron		✓
isMuon	if the particle is a muon		✓
isChargedHadron	if the particle is a charged hadron		✓
isNeutralHadron	if the particle is a neutral hadron		✓
isPhoton	if the particle is a photon		✓

Table 4.2: Input variables used in the Top quark tagging task (TOP) and the Quark-gluon tagging task (QG) with PIDs information.

Consequently, follow the feature-extraction method proposed by the authors, the data augmentation technique proposed in Chapter 3 is equipped with the `feature_transformation` method.

The training of ParticleNet consists in a maximum of 35 epochs, using an early stopping strategy based on the validation loss. The Adam optimizer is employed with a weight decay of 0.0001, along with the negative log-likelihood loss function. The initial learning rate is set to 3×10^4, and a reduce-on-plateau scheduler is used. The training is conducted with a batch size of 256.

4.2.2 LorentzNet implementation

The implementation of the LorentzNet model follows the one officially published by the authors in [19]. The network is implemented with PyTorch. It consists in 6 Lorentz group equivariant blocks ($L = 6$). The scalar embedding is implemented as one fully connected layer which maps the inputs to latent space of width 72, i.e., $Linear(scalar\_num, 72)$. The neural network ϕ_x, ϕ_e and ϕ_h, discussed in 3.2.2, are implemented as $Linear$(72,72) $\rightarrow$ *ReLU* $\rightarrow$ $BatchNorm1D$(72) $\rightarrow$ $Linear$(72,72). The neural network ϕ_m is implemented as $Linear$(72,1) $\rightarrow$ *Sigmoid*. The decoding layers are $Linear$(72,72) $\rightarrow$ *ReLU* $\rightarrow$ $Linear$(72,2). The dropout rate is set to be 0.2. The hyperparamater c in equation 3.8 is chosen to be 5×10^{-3} for the Top quark tagging task and 1×10^{-3} for the Quark gluon tagging task.

The input required by the model is the 4-momenta vector, alongside the PIDs information in the case of the Quark gluon tagging task. Consequently, the data augmentation technique proposed employs the `feature_transformation` method as the Identity function $\mathbb{I}$.

The training of LorentzNet is performed for 35 epochs. Firstly a linear warm-up period of 4 epochs is applied to reach the initial learning rate 1×10^{-3}. Then a $CosingeAnnealingWarmRestarts$ learning rate schedule with $T_0 = 4$, $T_{mult} = 2$ is adopted for the next 28 epochs. Finally, an exponential learning rate decay with $\gamma = 0.5$ is used for the last 3 epochs. The Adam optimizer is employed with a weight decay of 0.01 to minimize the cross-entropy loss. The training is conducted with a batch size of 128.

Chapter 5

Results

Chapter 5 presents the results of the experiments carried out in this master's thesis. After presenting the baseline used as a reference for the improvements obtained, the impact of the developed data augmentation technique on ParticleNet and its subsequent application on LorentzNet is shown.

5.1 The baseline

In this section of the results obtained from the master's thesis, it will be presented the initial accuracy baseline and the models' response to the invariance analysis when trained without implementing the data augmentation technique. All the results reported for each configuration are obtained by averaging the performance of 3 runs.

Both ParticleNet and LorentzNet are trained to perform the Top quark tagging task and the Quark-gluon tagging task. Concerning the latter, the dataset contains PIDs features. The values of accuracy obtained setting the baseline for the following experiments and their relative training time are reported in Table 5.1.

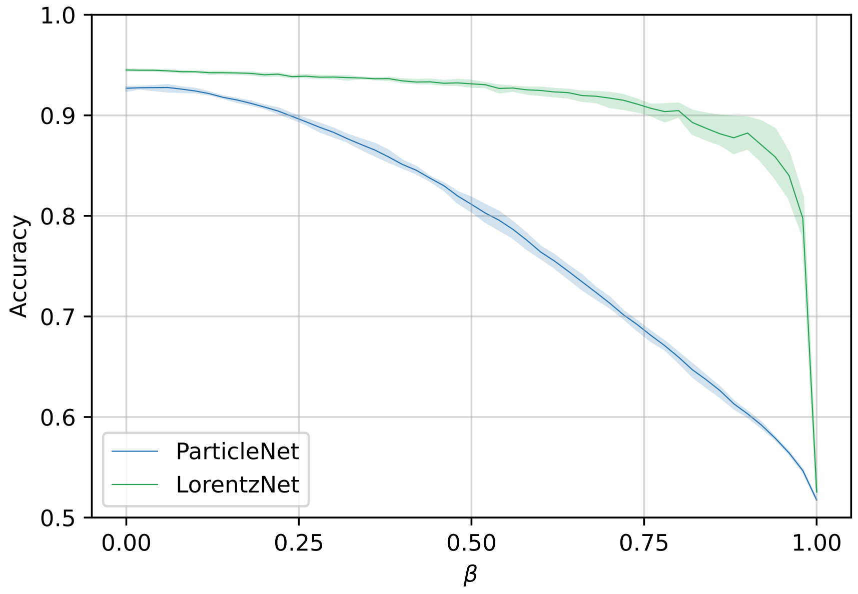
Based on the findings of this initial phase, it is evident that the LorentzNet model outperforms ParticleNet as expected. As described by the authors [19],

model	**Top quark tagging**	**Quark-gluon tagging**	**training time**
ParticleNet	0.929	0.819	**3.6 hours**
LorentzNet	**0.945**	**0.857**	17.2 hours

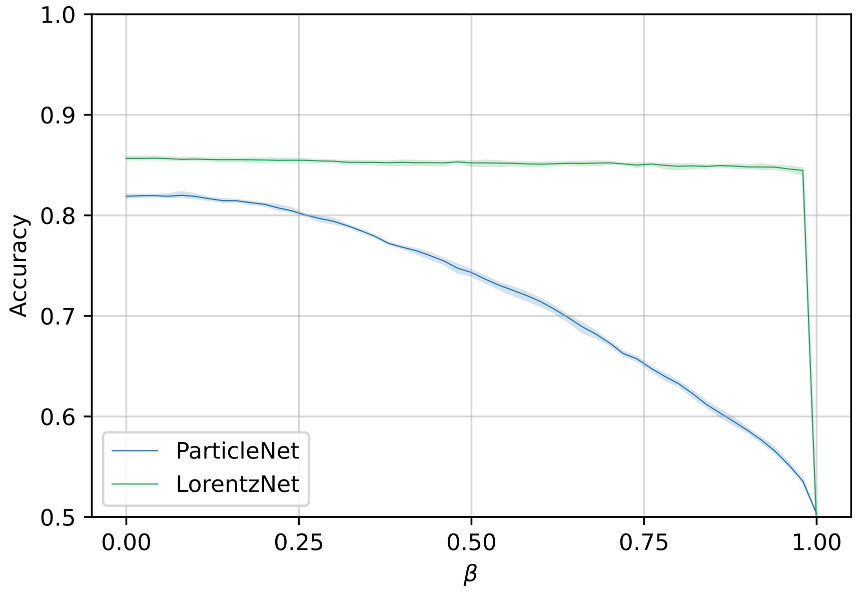
Table 5.1: The table shows the accuracy baselines for the experiments proposed. The values are obtained by averaging the performance of three runs.

LorentzNet is indeed able to achieve better tagging accuracy on both datasets. However, the main focus is not to align the performance of the two models on untransformed datasets. Instead, it aims to increase ParticleNet's awareness of the physical property of Lorentz invariance and align the models' performance in the invariance analysis. This specific goal enables the acquisition of a more robust and replaceable version of ParticleNet to LorentzNet, while retaining the advantage of being lighter and faster in inference.

The outcomes of the invariance analysis of two models not trained with augmentation are subsequently detailed, with the comparison between ParticleNet and LorentzNet on the identical datasets presented in a single figure. Figure 5.1 (a) illustrates the robustness of the two models in the Top quark tagging task, through an evaluation on the test set transformed at increasing beta values, as detailed in Chapter 3. Figure 5.1 (b) showcases the robustness of both models in the Quark-gluon tagging task, where the dataset is equipped with PIDs features. The obtained results demonstrate the anticipated patterns. As claimed, LorentzNet is more robust than ParticleNet in both experiments, but it is worth mentioning that the addition of the PIDs features in the case of the Quark-gluon tagging task makes Particlenet slightly more robust to invariance and LorentzNet significantly more performant in the analysis conducted.



(a)



(b)

Figure 5.1: The plots show the comparison between the robustness of ParticleNet (blue) and LorentzNet (green) to the Lorentz invariance in the Top quark tagging task (a) and Quark-gluon tagging task (b). The results are obtained by averaging the outcomes of three runs.

5.2 The data-augmentation impact

This section will showcase the outcomes achieved by applying the proposed data augmentation technique in the model training phase. The results will be quantified via model accuracy on the test set and invariance analysis, scrutinizing the experimental outcomes one by one.

To assess the impact of applying the data augmentation technique in the training phase, it was decided to train ParticleNet with augmentation at different values of β_{max}. Recalling that β is a parameter that ranges from 0 to 1, where zero indicates no augmentation, to cover the entire range, the chosen β_{max} values are 0.25, 0.50, 0.75, and 1. The base model is indicated as ParticleNet-base, while the augmented model are denoted as ParticleNet-βaug. The model performance is then compared to LorentzNet one, concluding with the train and evaluation of the LorentzNet model with $\beta_{max} = 1$ augmentation, to assess the benefit of the developed technique to the already invariance aware architecture.

5.2.1 Top quark tagging task

Presenting the results attained for the Top quark tagging task, Table 5.2 displays the accuracy achieved on the test set for ParticleNet trained with diverse augmentation levels, alongside the respective training time. It can be seen that the application of data augmentation does not noticeably affect the accuracy results obtained, even when applied at maximum beta, when the dissimilarity between the distributions of the untransformed training data and the test data becomes more pronounced. Furthermore, the model achieved its best accuracy value when the augmentation was applied at $\beta = 0.75$, indicating the benefit to the model of injecting the physical property of invariance and reinforcing the idea that the transformation at $\beta = 1$ is equivalent to imposing extreme conditions that the model can hardly interpret.

In addition, Figure 5.2 (a) presents the findings of the conducted invariance analysis. As can be seen, the augmented models near $\beta = 0$ show comparable performance to the base model. However, the focus is on values $\beta > 0$. When $\beta > 0.1$, the baseline model begins a drastic degradation of performance. In contrast, all augmented models consistently and largely outperform the base ParticleNet, with accuracy remaining at the highest level until β_{max} and even beyond. The augmented models exhibit an expected degradation in performance at different β values, coherently with their own training β_{max}. The degradation along the β axis is observed quite beyond the β_{max} value for each model, indicating that they can generalize further with respect to the training range. Finally, it is worth noting a different pattern for the 1.00-augmented model. In contrast to other models, this configuration shows degradation in performance before reaching $\beta = 1$, i.e., its β_{max}, as it is intuitively expected since extreme conditions requires specific solutions.

model	**accuracy**	**std**
ParticleNet-base	0.926	0.002
ParticleNet-0.25aug	0.926	0.002
ParticleNet-0.50aug	0.927	0.002
ParticleNet-0.75aug	**0.928**	0.001
ParticleNet-1.00aug	0.925	0.002

Table 5.2: The table shows the performance measured on the Top quark tagging test set for ParticleNet trained with diverse augmentation levels. The values are obtained by averaging the results of three runs.

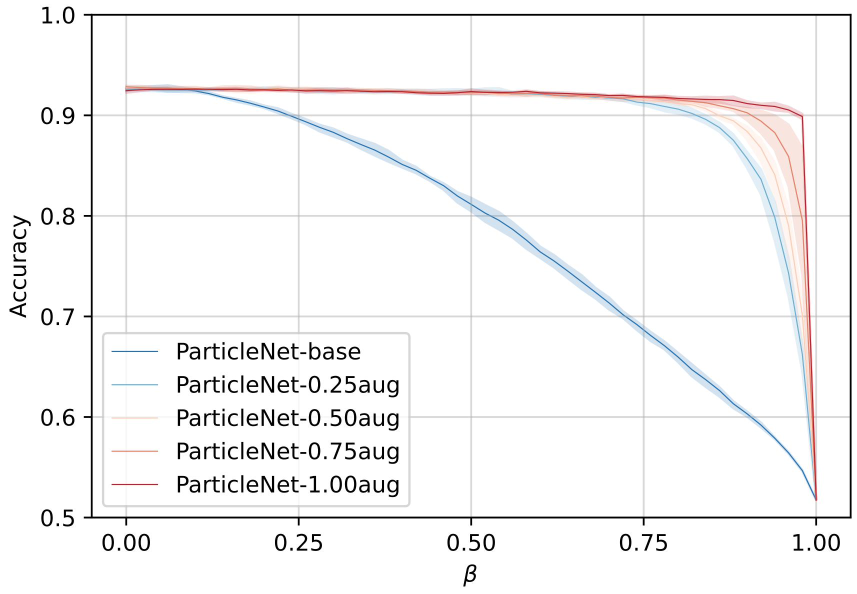
5.2.2 Quark-gluon tagging task

To present the results obtained for the quark-gluon tagging task, Table 5.3 shows the accuracy achieved on the test set for ParticleNet, which was trained with different levels of augmentation. The results show that the technique did not affect the obtained accuracy, even at maximum beta. The accuracy values remained stable when compared to the previous experiment, with a slight decrease at $\beta = 1$.

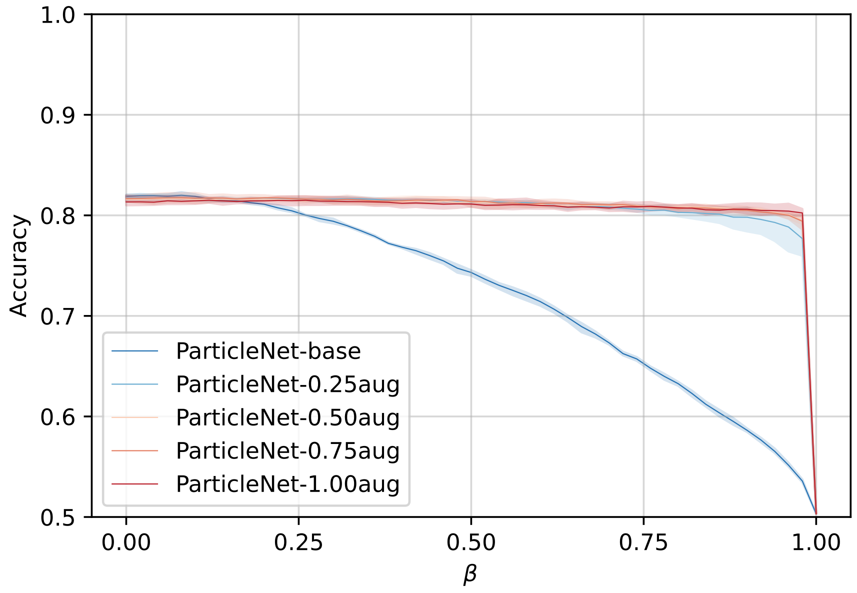
Additionally, the results of the invariance analysis are shown in Figure 5.2 (b). As can be observed, the augmented models near $\beta = 0$ show lower but still comparable performance to the base model. When $\beta > 0.1$ the augmented models outperform the base ParticleNet, showing the same trend of the Top quark tagging task. Nevertheless, in this particular case, the model tends to be more robust when it is already trained with the minimum β transformation, showing a degradation in performance at values well above its training β_{max}. Introducing additional features like PIDs in contrast to solely four-momentum, arguably, makes the model more robust initially, thus amplifying the effects of data augmentation.

model	**accuracy**	**std**
ParticleNet-base	**0.819**	0.002
ParticleNet-0.25aug	0.817	0.003
ParticleNet-0.50aug	0.816	0.001
ParticleNet-0.75aug	0.817	0.004
ParticleNet-1.00aug	0.813	0.005

Table 5.3: The table shows the performance measured on the Quark-gluon tagging test set for ParticleNet trained with diverse augmentation levels. The values are obtained by averaging the results of three runs.



(a)



(b)

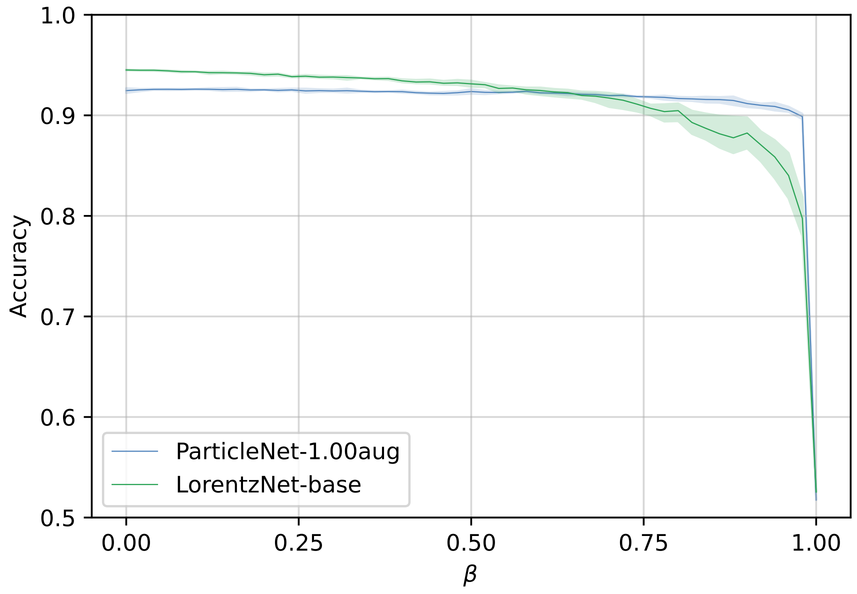
Figure 5.2: Accuracy at different values of β for ParticleNet models trained with different β_{max} values using Lorentz augmentation in the Top quark tagging task (a) and in the Quark-gluon tagging task (b). The results are obtained by averaging the outcomes of three runs.

5.2.3 The comparison with LorentzNet

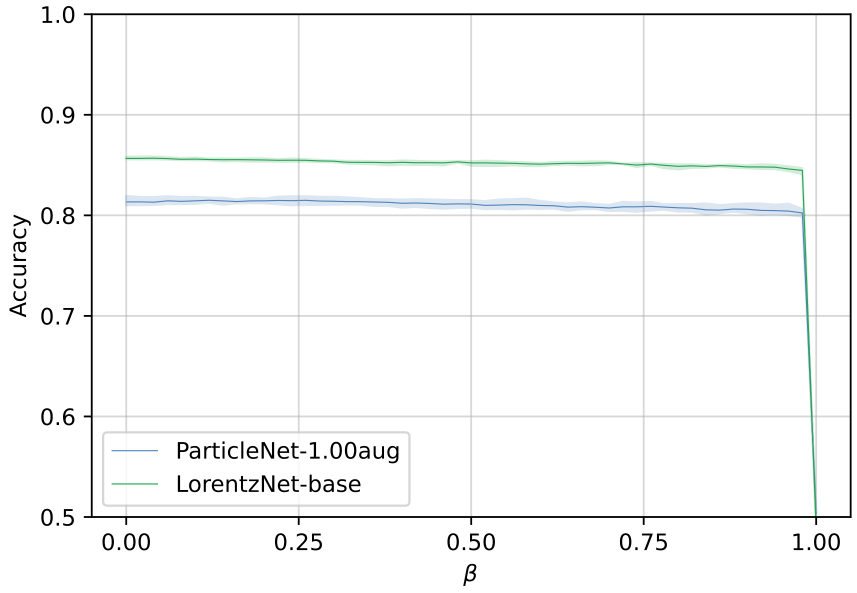
After discussing the results of data augmentation and its impact on the model's ability to be invariant to Lorentz transformations, now are here compared ParticleNet's new property directly with LorentzNet's existing capability. In conducting this analysis, ParticleNet-1.00aug was selected as the reference model. Although it does not perform best in terms of accuracy on the untransformed test set, it was found to be the most robust to Lorentz invariance in both the Top quark tagging task and the Quark-gluon tagging task.

Figure 5.3 (a) shows the comparison between the two model on the Top quark tagging task. LorentzNet outperforms ParticleNet for β values below 0.5. However, within the 0.5 to 0.75 range, ParticleNet demonstrates superior performance, indicating more stability against transformations at high β values.

Figures 5.3 (b) illustrates the comparison between the two model on the Quark-gluon tagging task. The observation that LorentzNet outperforms ParticleNet in this task is reaffirmed herein, as the former handles nearly the entire spectrum of transformations. However, ParticleNet demonstrates the same trend as LorentzNet, starting from a lower accuracy value. Therefore, the success of data augmentation in this experiment can also be affirmed.



(a)



(b)

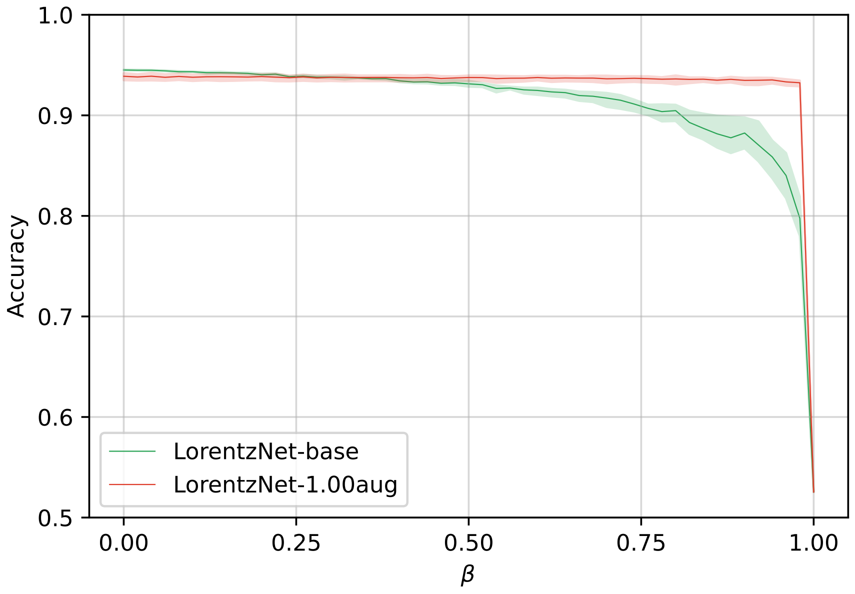
Figure 5.3: The plots show the comparison between the robustness of ParticleNet-1.00aug (blue) and LorentzNet (green) to the Lorentz invariance in the Top quark tagging task (a) and Quark-gluon tagging task (b). The results are obtained by averaging the outcomes of three runs.

5.2.4 Augmented LorentzNet

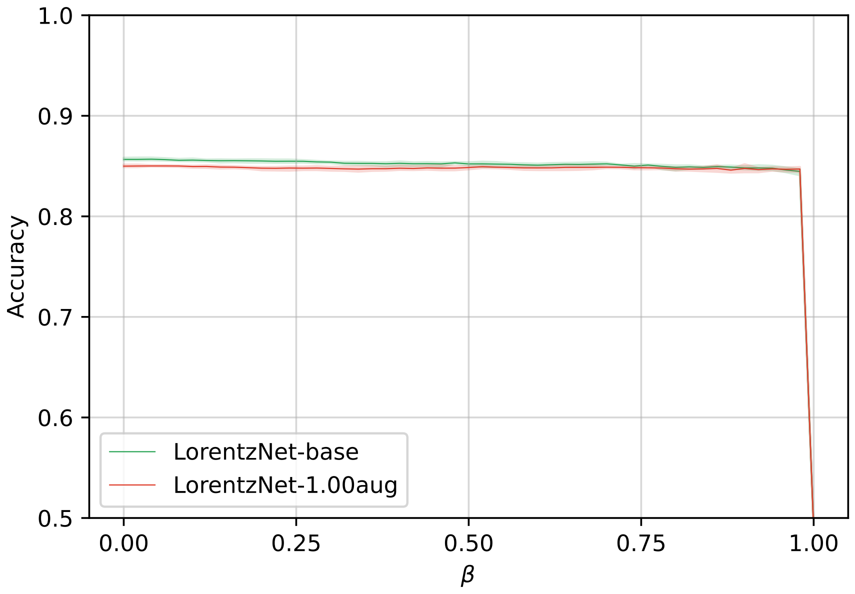
In this analysis, LorentzNet is trained using the data augmentation technique with $\beta = 1$. The base model is indicated as LorentzNet-base, while the augmented model is denoted as LorentzNet-1.00aug. The aim is to verify and quantify the impact of the aforementioned technique on a model that is already robust to invariance analysis.

Figure 5.4 (a) displays the results obtained in the Top quark tagging task. The use of data augmentation technique has a considerable impact. It is observed that the base model performs slightly better at β under 0.25, but a marked performance downturn is elicited thereafter. In contrast, the augmented model is robust across the entire β spectrum, outmatching LorentzNet-base, and exhibiting consistent performance up to the approach of betas to one. Therefore, it can be argued that despite the awareness of the invariance property within the architecture, the implementation of the data augmentation technique enhanced the model's performance by fully utilizing the neural network's capacity to learn. This outcome is inconsistent with LorentzNet's assertion of already being equivariant by design. Nevertheless, the model's architecture ought not be impacted by the introduction of data augmentation. Considering the experiment as an adversarial attack conducted via data augmentation, the initial reduction in performance followed by improvement with the increasing β augmentation indicates the necessity for further analysis on the LorentzNet equivariant block and its actual operation in future works.

Figure 5.4 (b) shows the results obtained in the Quark-gluon tagging task. In this instance, the implementation of the data augmentation technique failed to enhance the model's pre-existing perfect robustness towards invariance analysis. It is evident that the augmented model exhibits slightly worse performance than its base model counterpart, with performance alignment observed for β greater than 0.75. This outcome is likely a result of the dataset's property of having PIDs features, which, alongside the 4-momenta vector, enables LorentzNet to distinguish jets more effectively, even when transformed. Consequently, the data augmentation technique acts as noise in the training stage, leading to decreased model performance. Also the findings of this experiment do not support the claims of LorentzNet. While the augmented and baseline models demonstrate a similar trend in the invariance analysis, the initial drop in performance was unexpected. Future researches will focus on exploring equivariant blocking and the impact of PIDs features on the model's performance.



(a)



(b)

Figure 5.4: Accuracy at different values of β for LorentzNet models trained without data augmentation and with β_{max} value equals to 1 in the Top quark tagging task (a) and in the Quark-gluon tagging task (b). The results are obtained by averaging the outcomes of three runs.

Chapter 6

Conclusions

This master's thesis work presents a novel theory-guided training strategy for jet tagging tasks using a model-agnostic data augmentation technique. The goal is to investigate whether a model without inherent invariance preservation, i.e., ParticleNet, can effectively capture the essential physical properties of a jet, regardless of its orientation or Lorentz boost, acquiring properties similar to an invariant aware architecture as it is LorentzNet. Furthermore, the same training strategy is applied to the already invariant model, to enforce its model-agnostic property and explore the benefit of introducing more domain knowledge into a fully domain-aware architecture. By using this approach, the potential to enable deep learning models to understand the underlying physics of jets independent of specific transformations has been explored.

The experiments performed on the Top quark tagging dataset and on the Quark-gluon tagging dataset show that even a minimal application of the data augmentation strategy leads to a significant improvement in the robustness of a model that lacks of invariance preservation without compromising performance. These results show that the proposed approach has the potential to increase performance and improve generalization capabilities of the model. The outcomes are different when the same training strategy is applied to an invariant aware architecture. The data augmentation strategy brings improvement in performance where the model can potentially perform better, while it acts as noise when there is no scope for improvement. Furthermore, considering the application of the data augmentation as an adversarial attack to LorentzNet, the obtained results are inconsistent with the model assertion of already being invariant by design.

As part of future research, the functioning of the Lorentz equivariant blocks will be better investigated, trying to explain the anomalous behavior obtained in the last experiment. Subsequently, it is intended to integrate these functional blocks into the ParticleNet architecture, resulting in a hybrid model that combines the property of invariance by design with ParticleNet's time and resource efficiency. Furthermore, it is planned to investigate in more detail the impact of the proposed technique of introducing domain knowledge through data augmentation in other physics application, aiming to exploit the ability to learn during the training phase of deep learning models. This will consolidate the effective applicability of the technique in other tasks related to the physics world and on different models.

Acknowledgements

I would like to thank Professor Daniele Apiletti for giving me the opportunity to start and develop a thesis project that reflects my personal interests and for the trust he has shown in me along the way.
I would like to express my gratitude to Simone Monaco, PhD student in the Department of Control and Computer Engineering, for his helpfulness, close cooperation and commitment to the success of this research. And particularly for welcoming the idea of a thesis that combined my passion for physics and my love for Data Science.

Bibliography

[1] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, “Theory-guided data science: A new paradigm for scientific discovery from data,” *IEEE Transactions on knowledge and data engineering*, vol. 29, no. 10, pp. 2318–2331, 2017.

[2] G. Bell, T. Hey, and A. Szalay, “Beyond the data deluge,” *Science*, vol. 323, no. 5919, pp. 1297–1298, 2009.

[3] M. Diligenti, S. Roychowdhury, and M. Gori, “Integrating prior knowledge into deep learning,” in *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pp. 920–923, IEEE, 2017.

[4] J. Xu, Z. Zhang, T. Friedman, and Y. Liang, “Gv d. broeck. a semantic loss function for deep learning with symbolic knowledge,” *arXiv preprint arXiv:1711.11157*, vol. 2, 2017.

[5] A. Daw, A. Karpatne, W. D. Watkins, J. S. Read, and V. Kumar, “Physics-guided neural networks (pgnn): An application in lake temperature modeling,” in *Knowledge Guided Machine Learning*, pp. 353–372, Chapman and Hall/CRC, 2022.

[6] R. Stewart and S. Ermon, “Label-free supervision of neural networks with physics and domain knowledge,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[7] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, *et al.*, “Interaction networks for learning about objects, relations and physics,” *Advances in neural information processing systems*, vol. 29, 2016.

[8] K. Marino, R. Salakhutdinov, and A. Gupta, “The more you know: Using knowledge graphs for image classification,” *arXiv preprint arXiv:1612.04844*, 2016.

[9] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, “Robots that can adapt like animals,” *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.

[10] L. Von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, J. Pfrommer, A. Pick, R. Ramamurthy, *et al.*, “Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 614–633, 2021.

[11] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature communications*, vol. 5, no. 1, p. 4308, 2014.

[12] G. Karagiorgi, G. Kasieczka, S. Kravitz, B. Nachman, and D. Shih, "Machine learning in the search for new fundamental physics," *Nature Reviews Physics*, vol. 4, no. 6, pp. 399–412, 2022.

[13] G. Aad, X. S. Anduaga, S. Antonelli, M. Bendel, B. Breiler, F. Castrovillari, J. Civera, T. Del Prete, M. T. Dova, S. Duffin, *et al.*, "The atlas experiment at the cern large hadron collider," 2008.

[14] C. Collaboration, S. Chatrchyan, G. Hmayakyan, V. Khachatryan, A. Sirunyan, W. Adam, T. Bauer, T. Bergauer, H. Bergauer, M. Dragicevic, *et al.*, "The cms experiment at the cern lhc," *Jinst*, vol. 3, p. S08004, 2008.

[15] P. Keicher, "Machine learning in top physics in the atlas and cms collaborations," *arXiv preprint arXiv:2301.09534*, 2023.

[16] K. Datta and A. Larkoski, "How much information is in a jet?," *Journal of High Energy Physics*, vol. 2017, no. 6, pp. 1–25, 2017.

[17] H. Jiang, "Search for the lepton flavor violating decay z$\rightarrow$ eμ with the atlas detector at the lhc," 2019.

[18] H. Qu and L. Gouskos, "Jet tagging via particle clouds," *Physical Review D*, vol. 101, no. 5, p. 056019, 2020.

[19] S. Gong, Q. Meng, J. Zhang, H. Qu, C. Li, S. Qian, W. Du, Z.-M. Ma, and T.-Y. Liu, "An efficient lorentz equivariant graph neural network for jet tagging," *Journal of High Energy Physics*, vol. 2022, no. 7, pp. 1–22, 2022.

[20] G. Kasieczka, T. Plehn, A. Butter, K. Cranmer, D. Debnath, B. M. Dillon, M. Fairbairn, D. A. Faroughy, W. Fedorko, C. Gay, *et al.*, "The machine learning landscape of top taggers," *SciPost Physics*, vol. 7, no. 1, p. 014, 2019.

[21] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, "Jet-images—deep learning edition," *Journal of High Energy Physics*, vol. 2016, no. 7, pp. 1–32, 2016.

[22] A. collaboration *et al.*, "Quark versus gluon jet tagging using jet images with the atlas detector," *ATLAS Public Note ATL-PHYS-PUB-2017-017*, 2017.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition. corr abs/1512.03385 (2015)," 2015.

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision. 2015," *arXiv preprint arXiv:1512.00567*, 2015.

[25] D. Guest, J. Collado, P. Baldi, S.-C. Hsu, G. Urban, and D. Whiteson, "Jet flavor classification in high-energy physics with deep neural networks," *Physical Review D*, vol. 94, no. 11, p. 112002, 2016.

[26] C. collaboration *et al.*, "Cms phase 1 heavy flavour identification performance and developments," *CMS Detector Performance Summary CMS-DP-2017-013*, 2017.

[27] A. Butter, G. Kasieczka, T. Plehn, and M. Russell, "Deep-learned top tagging with a lorentz layer," *SciPost Physics*, vol. 5, no. 3, p. 028, 2018.

[28] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola, "Deep sets. corr abs/1703.06114 (2017)," *arXiv preprint arXiv:1703.06114*, 2017.

[29] Y. Wang, Y. Sun, Z. Liu, S. Sarma, M. Bronstein, and J. Solomon, "Dynamic graph cnn for learning on point clouds. corr," *arXiv preprint arXiv:1801.07829*, 2018.

[30] P. T. Komiske, E. M. Metodiev, and J. Thaler, "Energy flow networks: deep sets for particle jets," *Journal of High Energy Physics*, vol. 2019, no. 1, pp. 1–46, 2019.

[31] F. A. Dreyer, G. P. Salam, and G. Soyez, "The lund jet plane," *Journal of High Energy Physics*, vol. 2018, no. 12, pp. 1–42, 2018.

[32] A. Bogatskiy, B. Anderson, J. Offermann, M. Roussi, D. Miller, and R. Kondor, "Lorentz group equivariant neural network for particle physics," in *International Conference on Machine Learning*, pp. 992–1002, PMLR, 2020.

[33] J. Shlomi, P. Battaglia, and J.-R. Vlimant, "Graph neural networks in particle physics," *Machine Learning: Science and Technology*, vol. 2, no. 2, p. 021001, 2020.

[34] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[35] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[36] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*, pp. 1263–1272, PMLR, 2017.

[37] G. Kasieczka, T. Plehn, J. Thompson, and M. Russel, "Top quark tagging reference dataset," *Version v0 (2018_03_27). Mar*, 2019.

[38] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, and P. Z. Skands, "An introduction to pythia 8.2," *Computer physics communications*, vol. 191, pp. 159–177, 2015.

[39] J. De Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaitre, A. Mertens, and M. Selvaggi, "Delphes 3: a modular framework for fast simulation of a generic collider experiment," *Journal of High Energy Physics*, vol. 2014, no. 2, pp. 1–26, 2014.

[40] P. Komiske, E. Metodiev, and J. Thaler, "Pythia8 quark and gluon jets for energy flow," *Zenodo. doi*, vol. 10, 2019.

[41] W. Falcon and The PyTorch Lightning team, "PyTorch Lightning," Mar. 2019.

[42] L. Biewald, "Experiment tracking with weights and biases," 2020. Software available from wandb.com.

[43] J. Bardhan, A. Sinha, and K. Shah, "Particlenet pytorch geometric." `https://github.com/Jai2500/particlenet`, 2021.

[44] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.