Honors Report, Spring 2024

Sukhjinder Kumar (2020101055)

February 2024

Contents

1 Introduction

My work under Dr. Subhadip Mitra was on top tagging using physics-inspired deep learning models.

In particle physics, a jet refers to a collimated spray of particles that emerge from a high-energy collision between subatomic particles, typically protons or other hadrons. These collisions occur in particle accelerators such as the Large Hadron Collider (LHC) or in cosmic ray interactions with the Earth's atmosphere. Jets are produced when quarks and gluons, which are initially free particles, undergo a process called hadronization. This process results in the creation of color-neutral hadrons, such as protons, neutrons, and mesons, which then form the observed jet.

Jets are important phenomena in particle physics research because they provide crucial information about the underlying processes occurring in high-energy collisions. By studying the properties of jets, scientists can probe the fundamental interactions and particles that make up the universe. Jets are particularly relevant in the context of quantum chromodynamics (QCD), the theory that describes the strong force which binds quarks together to form hadrons.

Top tagging is a specific task in particle physics aimed at identifying jets that originate from the decay of a top quark (t-quark). The top quark is the heaviest known elementary particle and has a very short lifetime, decaying almost immediately after its creation. Because of its high mass, the top quark decays predominantly into a bottom quark (b-quark) and a W-boson (W^+ or W^-), which in turn can decay into lighter particles such as quarks, leptons, and neutrinos. The challenge in top tagging lies in distinguishing the jets arising from the decay of the top quark from other jets produced in the collision background.

State-of-the-art research in top tagging from a pure physics perspective often involves the development and application of advanced machine learning techniques to analyze the large amounts of data produced by particle physics experiments. These techniques include deep learning algorithms, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and graph neural networks (GNNs), which are trained on simulated and experimental data to classify jets as originating from top quark decays or from other sources.

I have gone through many papers, and below is a report based on 4 main papers that I have worked on this semester -

2 Paper 1: The Machine Learning Landscape of top taggers

1. **CNN (Convolutional Neural Network):**
   - **Use:** Applied to jet images generated from calorimeter data.
   - **Details:** Involves preprocessing steps like centering, rotating, and scaling based on the jet's properties to optimize the image for the

CNN. This approach is akin to techniques used in visual recognition tasks.

2. **ResNeXt:**
   - **Use:** A deep CNN variant used for analyzing jet images.
   - **Details:** Adapted to jet images with modifications in the convolutional layers to handle the unique data structure of jet images, such as different pixel intensities representing energy deposits.

3. **TopoDNN:**
   - **Use:** Based on topological data analysis of jet constituents.
   - **Details:** Inputs include p_T-sorted 4-vectors from jet constituents. It uses a dense network architecture and incorporates physics-informed preprocessing steps to align and scale data effectively.

4. **Multi-Body N-Subjettiness:**
   - **Use:** Utilizes N-subjettiness variables to capture the substructure of jets.
   - **Details:** These variables are derived from the angular distances between jet constituents and are used to feed a dense neural network, helping to differentiate signal from background effectively.

5. **TreeNiN (Tree-based Network in Network):**
   - **Use:** Operates on jet trees that represent hierarchical structures within jet constituents.
   - **Details:** This method employs a tree-based architecture where each node of the tree is processed using Network in Network layers to extract and classify features from jet constituents.

6. **P-CNN (Particle Convolutional Neural Network):**
   - **Use:** Applies CNNs directly to sequences of jet constituents.
   - **Details:** Unlike typical CNNs that process 2D images, P-CNN processes 1D sequences where each sequence element represents a jet constituent with multiple features.

7. **ParticleNet:**
   - **Use:** A network designed for point cloud data, which treats jets as unordered sets of particles.
   - **Details:** Utilizes dynamic graph convolution over the constituents to efficiently process and classify jets based on their substructure.

8. **Lorentz Boost Network (LBN):**

- **Use:** Extracts features by boosting constituents to appropriate rest frames.
- **Details:** This network computes linear combinations of input 4-vectors and uses Lorentz transformations to analyze features in the jet's rest frame, enhancing the physical interpretability of the features.

9. **Lorentz Layer (LoLa):**
   - **Use:** Extracts features by boosting constituents to appropriate rest frames.
   - **Details:** This network computes linear combinations of input 4-vectors and uses Lorentz transformations to analyze features in the jet's rest frame, enhancing the physical interpretability of the features.

10. **Latent Dirichlet Allocation (LDA):**
    - **Use:** Applied in a generative model framework to uncover latent structures in jet data.
    - **Details:** Traditionally used in text analysis, here it identifies underlying themes or patterns in jet substructures that correlate with physical processes.

11. **Energy Flow Polynomials (EFPs):**
    - **Use:** Forms a linear basis for IRC-safe observables using energy correlators.
    - **Details:** These polynomials capture energy correlations among jet constituents under different angular conditions, providing a comprehensive description of jet substructure.

12. **Energy Flow Networks (EFNs) and Particle Flow Networks (PFNs):**
    - **Use:** These networks encode symmetries and physics constraints directly into their architecture.
    - **Details:** EFNs and PFNs process data by transforming constituent features into a latent space and then summarizing them into a global event descriptor. This technique ensures that the networks are sensitive to the physical properties of the jets while being invariant to permutations of the inputs.

Each algorithm leverages unique aspects of the jet data and applies different strategies from machine learning to optimize the identification of hadronically decaying top quarks, demonstrating the richness and variety of approaches in the field of particle physics machine learning.

3 Paper 2: INVARIANT AND EQUIVARIANT GRAPH NETWORKS

The paper "Invariant and Equivariant Graph Networks" focuses on the characterization of permutation invariant and equivariant linear layers for graph data, particularly those defined on hypergraphs. It extends the understanding of how to construct neural networks that respect the underlying symmetries of their input data—symmetries like the reordering of nodes within a graph, which should not affect the output of the network if it is to be considered invariant or equivariant.

3.1 Key Concepts and Contributions

Invariant and Equivariant Properties:

Invariant: A function is invariant under some transformation if applying the transformation to the input does not change the output. For graph networks, an invariant function produces the same output regardless of the node ordering in the input graph.

Equivariant: A function is equivariant if transforming the input results in a similar transformation in the output. For graphs, this means if the nodes of the input graph are reordered, the outputs are reordered in the same way.

Characterization of Linear Layers:

The paper provides a theoretical framework to fully characterize all linear invariant and equivariant transformations possible on sets, graphs, and hypergraphs. This is crucial for designing graph neural networks that need to handle varying input sizes and complex patterns efficiently.

Dimensionality of Invariant and Equivariant Functions:

It establishes that the dimensionality of these functions for graphs can be determined using concepts from combinatorial mathematics (like Bell numbers), which count the ways to partition a set. These dimensions dictate the complexity and capability of the neural network layers to handle graph structured data.

Orthogonal Basis for Layers:

The study computes orthogonal bases for these invariant and equivariant layers, which simplifies the implementation of such layers in neural networks. This helps in constructing networks that are both theoretically sound and practically effective.

Universal Approximation Capability:

A significant theoretical result is that the models described can approximate any message-passing neural network. This positions invariant and equivariant graph networks as a powerful class of models capable of generalizing across different graph-based learning tasks.

3.2 Practical Application to Top Tagging

In the context of top tagging, a task in particle physics where one aims to identify particles known as top quarks from collision data, graph neural networks can play a crucial role. The data from these collisions can naturally be represented as graphs where nodes represent particle detections and edges may represent various physical properties or detected relationships.

Equivariance in Physics: Maintaining equivariance is particularly useful in physics as it allows the model to learn physical laws and relationships in the data that are independent of the reference frame or the labeling of inputs in the graph (e.g., detector outputs).

Handling Complex Data: The ability to handle hypergraphs and the established robustness against changes in input (like reordering of nodes) makes these networks particularly suited for the erratic and complex nature of particle collision data.

By utilizing invariant and equivariant graph networks, researchers can build models that better generalize across different experimental setups and observational conditions without losing accuracy in identifying and classifying particles like top quarks. This reduces the need for extensive retraining or model adjustments as experimental conditions evolve or new data is collected.

4 Paper 3: Jet Tagging via Particle Clouds

The paper "Jet Tagging via Particle Clouds" introduces an innovative approach to jet tagging using machine learning techniques, particularly focusing on a method called ParticleNet. Here's a detailed explanation of the main ideas discussed in the paper:

4.1 Jet Representation in Physics

Jets are collimated sprays of particles produced in high-energy processes such as proton-proton collisions at the Large Hadron Collider (LHC). Understanding the origin of these jets (which type of elementary particle initiated them) is crucial for studying the fundamental aspects of particle physics. Traditional methods have used quantum chromodynamics (QCD) to differentiate jets initiated by different types of particles (e.g., quarks vs. gluons), but these methods can be enhanced significantly using machine learning.

4.2 Particle Cloud Concept

The core idea proposed in the paper is to treat a jet as a "particle cloud" — an unordered set of its constituent particles. This representation is inspired by point clouds used in computer vision and is particularly suited for processing with machine learning algorithms because it naturally respects the permutation symmetry (the order of points does not matter).

4.3 ParticleNet Architecture

ParticleNet is a neural network architecture developed specifically for jet tagging. It employs Dynamic Graph Convolutional Neural Networks (DGCNN) to process particle clouds. Unlike traditional convolutions used in image processing, the dynamic graph convolutions are adaptable to the irregular data structure of particle clouds, capturing local and global relationships between particles.

4.4 Advantages of Particle Cloud Representation

Efficiency: Directly works with raw particle data without needing to impose any particular structure (like grids in images or sequences), making it computationally efficient and flexible.

Permutation Invariance: By treating the jet as an unordered set, the model does not require any predefined ordering of the particles, avoiding potential biases or inefficiencies from imposed orderings.

Improved Performance: The paper reports that ParticleNet achieves state-of-the-art performance on benchmark datasets for jet tagging, surpassing previous machine learning methods and traditional approaches.

4.5 Implementation and Evaluation

ParticleNet was evaluated on several benchmarks, including top quark tagging and quark-gluon tagging, showing significant improvements over existing methods. The network architecture can dynamically adapt during training, learning the most relevant features of particle interactions within jets.

4.6 Broader Implications

While the paper focuses on jet tagging, the approach has potential applications across various areas of particle physics where understanding complex, many-particle systems is crucial. It also demonstrates the power of machine learning in advancing our understanding and capabilities in experimental physics.

5 Paper 4: "An Efficient Lorentz Equivariant Graph Neural Network for Jet Tagging"

The paper "An Efficient Lorentz Equivariant Graph Neural Network for Jet Tagging" presents a detailed study on a deep learning model, LorentzNet, designed for jet tagging in particle physics. Below is a comprehensive breakdown of the paper including the model architecture, input handling, and the underlying physics guiding the model's design:

5.1 Model Overview

LorentzNet is developed to enhance jet tagging efficiency by preserving Lorentz symmetry, a fundamental spacetime symmetry for elementary particles. Traditional models often fail to maintain this symmetry, leading to inefficiencies. A neural network layer is Lorentz-equivariant if its output transforms accordingly when the input undergoes a Lorentz transformation. Therefore, equivariant layers can be stacked to build a symmetry-preserving deep neural network, of which the classification result is unchanged under any Lorentz transformation. They have used a graph neural network.

Graph neural networks are natural to learn representations for graph-structured data. Given a graph $G = (V, E)$, assuming L steps in total, the l-th message passing step on the graph can be described as:

$$m_i^{l+1} = \sum_{j \in N(i)} M_l(h_i^l, h_j^l, e_{ij});$$

$$h_i^{l+1} = U_l(h_i^l, m_i^{l+1});$$

where $h_i^0 = f_i$ is the input feature, e_{ij} is the edge feature, $N(i)$ is the set of neighbors of the 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 $\{h_i^L; i \in [N]\}$.

5.2 Theory

Minkowski metric: Consider the 4-dimensional space-time $\mathbb{R}^4$ with basis $\{e_i\}_{i=0}^3$. We define a bilinear form $\eta : \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}$ as follows. For $u, v \in \mathbb{R}^4$, we set

$$\eta(u, v) = u^T J v$$

where $J = \mathrm{diag}(1, -1, -1, -1)$ 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: The Lorentz group is defined to be the set of all matrices that preserve the bilinear form η. Additionally, the Lorentz boost is also included. Given two inertial frames $\{e_i\}_{i=0}^3$ and $\{e'_i\}_{i=0}^3$, the relative velocity vector β and the boost factor γ are defined by

$$e'_0 = \gamma e_0 + \sum_{i=1}^{3} \gamma \beta_i e_i$$

where $\gamma = (1 - \beta^2)^{-1/2}$. If we perform a Lorentz boost along the x-spatial axis, then the Lorentz transformation between these two frames is the matrix

$$Q = \begin{pmatrix} \gamma & -\gamma\beta & 0 & 0 \\ -\gamma\beta & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Group Action: A group action of a group G on a set S is a group homomorphism from G to some group (under function composition) of functions from S to itself. If G is a group with identity element e, and X is a set, then a (left) group action α of G on X is a function

$$\alpha : G \times X \to X,$$

that satisfies the following two axioms:

Identity:

$$\alpha(e, x) = x$$

for all $x \in X$.

Compatibility:

$$\alpha(g, \alpha(h, x)) = \alpha(gh, x)$$

for all $g, h \in G$ and all $x \in X$.

The group G is then said to act on X (from the left). A set X together with an action of G is called a (left) G-set.

Lorentz group equivariance: Let $T_g : V \to V$ and $S_g : U \to U$ be group actions of $g \in G$ on sets V and U, respectively. We say a function $\varphi : V \to U$ is equivariant to group G if

$$\varphi(T_g(v)) = S_g(\varphi(v))$$

holds for all $v \in V$ and $g \in G$.

Proposition: A continuous function $\varphi : (\mathbb{R}^N \times 4) \to \mathbb{R}^4$ is Lorentz-equivariant if and only if

$$\varphi(v_1, v_2, \ldots, v_N) = \sum_{i=1}^{N} g_i \left(\langle v_i, v_j \rangle_{i,j=1}^{N} \right) v_i,$$

where g_i are continuous Lorentz-invariant scalar functions, and $\langle \cdot, \cdot \rangle$ is the Minkowski inner product.

Above Proposition provides a way to construct Lorentz group equivariant mappings without the need to calculate high-order tensors (as done in LGN paper). Instead, a Lorentz group equivariant continuous mapping can be constructed by focusing on v_i with encoding the Minkowski dot products of v_i with its neighbors. This motivates us to design the Minkowski dot product attention in LorentzNet.

5.3 Input

The input to LorentzNet includes the 4-momenta of particles generated in high-energy collisions (x_i) and scalar input (h_i). The scalar input's embedding is treated as the nodes of a graph where particles have edges that facilitate message passing, reflecting the interaction between particles.

5.4 Why does it work?

As the update steps for x are Lorentz invariant by design (based on proposition) they remain Lorentz invariant. As for scalars, they are updated based on prior scalars (by definition Lorentz invariant, just scalars) and the dot product of Lorentz vectors which is also an invariant (a constant). Hence both x and h are Lorentz invariant throughout. We know from prior works that graph neural networks are able to learn top tagging classification with the added benefit of being permutation invariant. This paper uses that power and adds a layer of Lorentz invariance on top of it.

5.5 Model Architecture

Lorentz Group Equivariant Block (LGEB):

It processes nodes (particles) and updates their states based on the Minkowski inner products of their 4-momenta, leveraging the Lorentz symmetry.

Message passing within the graph is designed to be Lorentz equivariant. This is achieved using scalar functions of the Minkowski dot products between different particle momenta, which guide the network in aggregating information across the graph.

When $l = 0$, x_i^0 equals the input of the 4-momenta v_i 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.

We use m_{ij} to denote the edge message between particle i and j, and it encodes the scalar information of the particle i and j, i.e.,

$$m_{ij} = \varphi_e(h_{li}, h_{lj}, \psi(k \cdot x_{li} - x_{lj} \cdot k^2), \psi(\langle h \cdot x \rangle_{li,lj})),$$

where $\varphi_e(\cdot)$ is a neural network and $\psi(\cdot) = \mathrm{sgn}(\cdot)\log(|\cdot| + 1)$.

Attention Mechanism:

The model incorporates a novel attention mechanism called Minkowski dot product attention. This component computes attention weights based on Lorentz invariant geometric quantities derived from the Minkowski metric, helping to focus on relevant features in the jet.

In line with the Proposition defined in the theory section, we update Lorentz vectors as - We design Minkowski dot product attention as:

$$x_i^{l+1} = x_{li} + c \cdot \sum_j \varphi_x(m_{ij}) \cdot x_{lj}$$

The scalar features for particle i are forwarded as:

$$h_i^{l+1} = h_{li} + \varphi_h \left(h_{li}, \sum_{j=1}^{N} w_{ij} m_{ij} \right),$$

where $\varphi_h(\cdot)$ is also modeled by neural networks whose output dimension equals the dimension of h^{l+1}. For efficient computation, we operate summation

$\sum_{j=1}^{N} w_{ij} m_{ij}$. We introduce a neural network $\varphi_m(\cdot)$ to learn the edge significance from node j to node i, i.e., $w_{ij} = \varphi_m(m_{ij}) \in [0, 1]$. This can both ensure the permutation invariance and also ease the implementation for jets with different numbers of particles. This operation has been widely adopted in other types of graph neural networks

Equivariance to the Lorentz Group:

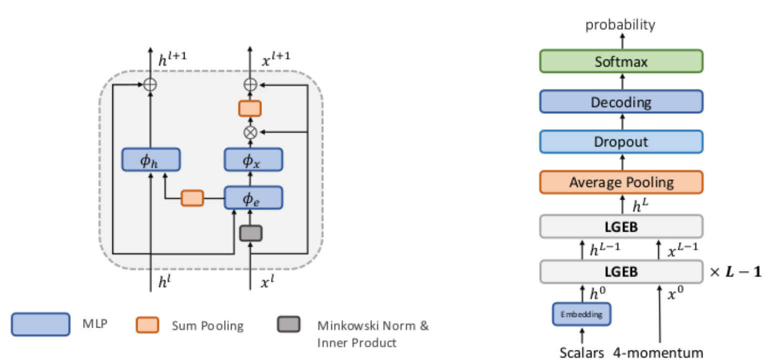
The architecture ensures that the outputs transform correctly under Lorentz transformations of the inputs, thereby preserving the physical symmetries inherent in the data.

Proposition 3.2. The coordinate embedding $x^l = (x_1^l, x_2^l, \ldots, x_N^l)$ are Lorentz group equivariant and the node embedding $h^l = (h_1^l, \ldots, h_N^l)$ are Lorentz group invariant.

Proof: We denote Q as the Lorentz transformation. If m_{ij}^l are invariant under Q for all i, j, l, x_i^{l+1} will be Lorentz group equivariant because

$$\begin{aligned} Qx_i^{l+1} &= Q(x_i^l + c \cdot x_j^l \cdot \varphi_x(m_{ij})), \\ &= Qx_i^l + c \cdot \sum_{j=1}^{N} Qx_j^l \cdot \varphi_x(m_{ij})). \end{aligned}$$

Then we illustrate the invariance of m_{ij}^l. Let's start from the input. Since Minkowski norm and Minkowski inner product are invariant to Lorentz group, we have $\|x_i^0 - x_j^0\|^2 = \|Qx_i^0 - Qx_j^0\|^2$ and $\langle x_i^0, x_j^0 \rangle = \langle Qx_i^0, Qx_j^0 \rangle$. Therefore, the input of φ_e are invariant variables under transformation Q and then m_{0ij} are invariant. Recursively using the invariance of m_{ij}^l and the equivariance of x_i^l, we can get the conclusion.



Lorentz Group Equivariant Block (LGEB) LorentzNet

5.6 Inference

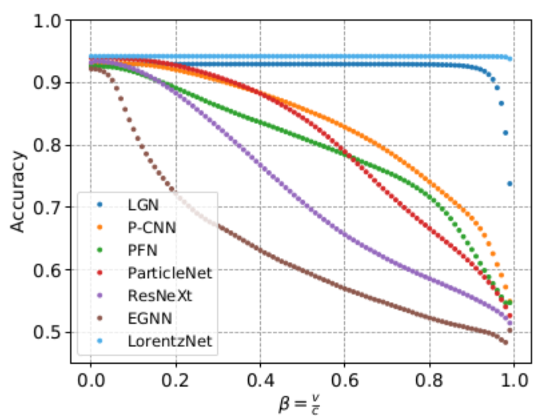
To do the inference, we consider only the node embeddings (h_i^L), as the "information" content of 4-vectors by virtue of message parsing would have been encoded in the node. We take all the h^L vectors and compute their average. We then pass this weighted aggregate into a neural network binary classifier (a 2-layer dense neural network followed by a softmax to get probabilities if it belongs or not to the class) to find if it is the required jet or not. (*A subsequent dropout layer is applied to h^{av} prior to classification to prevent overfitting)

5.7 Implementation

The LorentzNet architecture used in this paper is shown in Figure. It consists of 6 Lorentz group equivariant blocks ($L = 6$). The scalar embedding is implemented as one fully connected layer which maps the scalars to a latent space of width 72, i.e., `Linear(scalar_num, 72)`. φ_x, φ_e, φ_h are all implemented as `Linear(72, 72) -> ReLU -> BatchNorm1d(72) -> Linear(72, 72)`. φ_m is implemented as `Linear(72, 1) -> Sigmoid`. The decoding layers are `Linear(72, 72) -> ReLU -> Linear(72, 2)`.

5.8 Equivariance Test

They tested the model trained on the original training data, and the tagging accuracy on the rotated test data is reported in the below figure. The horizontal axis of shows the value of β, and the vertical axis shows the tagging accuracy on the top tagging dataset under Lorentz transformation with the corresponding β. The results show that the accuracy of LorentzNet and LGN on the test data after Lorentz transformation is robust in a large range of β, while the test accuracy of other non-equivariant models will drop as β becomes larger. According to special relativity, the fundamental quantities to clarify the particles will not be changed.



5.9 Computational Efficiency

Unlike previous models that required the computation of high-order tensors, LorentzNet simplifies the computational burden by focusing on first-order interactions and employing efficient attention mechanisms. This results in faster training and inference times. The authors have also provided a smaller model

(LorentzNet small) that has a fraction of the size and performs comparatively). To reduce the size, they have used 16 as the latent dimension for embeddings, and thus all corresponding neural networks that output similar dimensions are shrunk. The inference time is also reduced by a factor of 2 on the GPU and 13 on the CPU. The accuracy is less by 1 part in 1000! False acceptance and rejections are also very comparable.

5.10 Future Work

The paper uses hand-crafted equations for updating co-ordinates and message parsing, in the proven graph neural network methodology. They have created equations s.t Lorentz invariance is preserved. However, in practice, it is an overkill, even for collider events. We usually need Lorentz invariance in only one to two degrees of freedom (out of possible 6, which the paper accomplishes). Hence we can loosen up the equation to allow for weaker invariance and in the process improve computational efficiency, and accuracy. Some of the other invariances include (but are not limited to): x-y rotation invariance, IRC safe (infrared and collinear safe), Poincare invariance (Lorentz + translation), and any group transformation!