Graph Neural Networks in Particle Physics

Akanksha Ahuja [1] Michele Selvaggi [2] Loukas Gouskos [2] Daniela Bortoletto [1,2]

[1]University of Oxford [2]CERN

Introduction

Our objective is to perform **final state particle classification,** based on the particle's origin of decay in the event of two leptons (electron and positron) **colliding at the FCC-ee detector**, a future circular detector, proposed at CERN in the coming decades. Based on the classification of the final state, as **decaying from the Higgs Boson, or Z Boson,** we group the particles of the same class together.

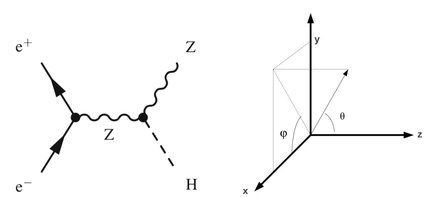

Figure 1. (Higgsstrahlung Process (Left), Reference Frame (Right)

Background

Jets are a collimated spray of **stable particles, which are directly observed in an experiment.** Jets are first clustered and then classified into specific types (jet tagging or jet classification) to understand and identify the origin of the stable observed particles. Our purpose is to cluster particles and specify which set of final state (stable) particles were generated from the decay of the Higgs Boson, Z boson or other types of particles.

Research Questions

- Could **machine learning techniques** be used to predict the sets of final state particles originating from the Higgs Boson and Z boson in the HZ production mode at the future lepton collider?
- How can **graph neural networks be used to model event collisions** as graphs and particles as nodes for the supervised task of node classification?
- What is the **most optimal type of graph data representation** of the stable particles detected at a collider?
- What is the **most promising graph neural architecture** to correctly predict all the node's labels in a graph?

Node Features	Physical significance
PID	Process ID
pos_r	position: vertex 4-vector(distance from the centre)
pos_theta	position:vertex 4-vector(polar angle)
pos_phi	position:vertex 4-vector(azimuth angle)
pos_t	position:vertex 4-vector(time)
mom_p	momentum 4-vector(transverse momentum)
mom_theta	momentum 4-vector(polar angle)
mom_phi	momentum 4-vector(azimuth angle)
mom_mass	momentum 4-vector(mass)

Table 1. Node Features of the Dataset

Methodology

Our research project builds a model for **supervised node classification** for jet clustering in an electron-positron collision within an **inductive learning** setup on a **simulation dataset.** Each event **collision is represented as a graph**, whereas the **final state particles (observed at the detector) are represented as nodes.** In our graph $G = (V, E)$, the objective is **to classify each vertex** v **belonging to a set of nodes** V **into the correct class.** The dataset includes multiple graph G and a list of pairs of nodes and their corresponding labels, depicting their category. The desired outcome is a **node-level prediction of the particle class:** whether it decayed from an H boson, Z boson, or other particles in an $e + e-$ collision.

Graph Generation Strategies

- **edge-KNN:** We connect the k-nearest neighbours (k=8) in an event.
- **edge-radius:** After dimensionality reduction of the dataset into a 2-D space, we specify a distance metric defined as radius threshold (r=0.2), and we connect all the particles within a specific radius together.
- **edge-label** We connect all the nodes of the same class in an event (or having the same labels) to assess the performance difference by comparing it with edge-radius and edge-knn.

Graph dataset	Number of nodes*	Number of edges
edge-radius	60	3540
edge-knn	60	480
edge-label	60	3368

Table 2. A sample event is used to illustrate the characteristics. *Number of nodes vary for each event

Graph Neural Networks

A graph neural network employs a type of **neural message passing** [1]. In such a framework, **vector messages are passed between nodes and neural networks are used to update them**. We take an input graph $G = (V, E)$ with node features $\mathbf{X} \in \mathbb{R}^{d \times |\mathcal{V}|}$. Using the node features, we **generate an output of node embeddings:** $\mathbf{z}_u, \forall u \in \mathcal{V}$.

Figure 2. Message Passing Networks: Aggregation from a node's local neighbourhood [1]

A hidden embedding $\mathbf{h}_u^{(k)}$ of node $u \in \mathcal{V}$ is updated according to information collected from u's graph neighbourhood $\mathcal{N}(u)$ during each message passing step in a graph neural network. The **message-passing update is mathematically expressed such that the UPDATE function and AGGREGATE function** are both arbitrary differentiable functions (such as neural networks). Here $\mathbf{m}_{\mathcal{N}}(u)$ is the aggregated message from u's neighbourhood $\mathcal{N}(u)$ [1].

$$\mathbf{h}_u^{(k+1)} = \text{UPDATE}^{(k)}\left(\mathbf{h}_u^{(k)},\ \text{AGGREGATE}^{(k)}\left(\left\{\mathbf{h}_v^{(k)}, \forall v \in \mathcal{N}(u)\right\}\right)\right) \tag{1}$$

$$= \text{UPDATE}^{(k)}\left(\mathbf{h}_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right) \tag{2}$$

.

Results

We build **8 types of sequential graph neural network architectures** with **various network depths (2,4,8,16)** and **hyper-parameters (learning rate, weight decay, activation, hidden channels, dropout, optimiser)** and evaluate the results of our predictions at the **node-level accuracy** and the **event-level accuracy** against the **baseline neural network (MLP).**

Our key findings report an increased performance against the established baseline, the most suitable network depth, the best performing GNN architectures, graph processing scheme, and edge-generation scheme.

Model	Depth	Dataset	Event Accuracy	Node Accuracy
MLP	2	KNN	87.2637%	99.9538%
GCN	2	KNN	90.7206%	99.8618%
Cheb	2	KNN	95.4245%	99.9591%
SAGE	2	KNN	95.0667%	99.9518%
TAGCN	4	Radius	91.3637%	99.9611%
GAT	2	KNN	95.8284%	99.9414%
GIN	2	KNN	88.1157%	99.7083%
JK	2	KNN	86.3539%	99.9302%
SuperGAT	4	KNN	96.4912%	99.9494%

Table 3. GNN-variants: Model Evaluation and Comparison

Key Findings

- The majority of the GNN **models perform better on the edge-KNN dataset**. We generated three types of datasets (edge-KNN, edge-Radius, edge-Label); the edge-knn dataset is the most suitable representation. Almost all models perform well on it in comparison to edge-radius and edge-label. Further, in terms of graph processing, **the model performs better on variable-sized graphs** in comparison to the fixed-size graphs.
- Baseline: **MLP-2 event accuracy is 87.2637%** and **GCN-2 is 90.7206%** on the edge-KNN dataset. As can be observed, the graph convolutional network outperformed neural networks. After extensive investigations, we uncovered **network depths of 2 and 4 are the most suitable** for the specified task of supervised node classification.
- The best performing model is built by **stacking 4 layers of superGAT's convolutional operator.** SuperGAT-4 outperforms other GNN networks and reports an **event accuracy of 96.4912%** on the edge-KNN dataset. We implement and optimise various GNN architectures: GAT, SAGE, GIN, JKNet, ChebNet, TAGCN, superGAT, and see improvement across all models against MLP baseline on the event accuracy metrics.

References

[1] William L. Hamilton.
Graph representation learning.
Synthesis Lectures on Artificial Intelligence and Machine Learning, 14(3):1–159.

[2] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant.
Graph neural networks in particle physics.
Machine Learning: Science and Technology, 2:021001, 01 2021.

[3] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia.
Learning to simulate complex physics with graph networks.
arXiv:2002.09405 [physics, stat], 09 2020.

[4] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun.
Graph neural networks: A review of methods and applications.
AI Open, 1:57–81, 2020.

akanksha.ahuja99@gmail.com