GOAL -

The goal of this task is to classify high-energy particle jets as being either **quark** or **gluon-**initiated. The dataset consists of jet information in point-cloud data, where each event contains multiple particles with features such as transverse momentum (pT), rapidity, azimuthal angle, and energy.

Since particle jets have inherent relational structures, using **Graph Neural Networks (GNNs)** is an effective approach. Instead of treating the dataset as flat arrays, I converted it into **graphs**, where each particle is represented as a node, and edges are formed based on their relationships.

IMPLEMENTATION-

I implemented and compared two GNN architectures:

1. ParticleNet (EdgeConv-based GNN)
2. Graph Isomorphism Network (GIN)

Before feeding the data into GNN models, the raw point-cloud data (particles inside jets) must be converted into a graph structure. Each particle in a jet becomes a node, and edges are formed based on similarity using **k-nearest Neighbors (KNN)**.

1. **Node Features:** Each particle in a jet is described by:

    a) **Transverse momentum (pT)**
    b) **Rapidity**
    c) **Azimuthal angle**
    d) **Energy**
2. **Edge Construction:** I constructed edges using KNN to connect each node to its k-nearest neighbors based on spatial distance.

3. **Graph Representation:**

    a) The node features remain unchanged.
    b) The edge index stores the connections between nodes.

These graphs are then used as input for GNN-based architectures.

1st Model: ParticleNet (EdgeConv-based GNN)

Architecture

ParticleNet is inspired by the **Dynamic Graph CNN (DGCNN)** proposed by *Yue Wang et al. (2019),* which introduced EdgeConv, a graph convolutional operation designed for point-cloud learning **[1]**.

Unlike traditional graph convolutions, which operate on fixed edges (pre-defined based on heuristics), EdgeConv dynamically learns and updates edges at every layer, allowing the model to adaptively capture the most important relationships between particles. This dynamic connectivity is highly beneficial in jet physics, where interactions between particles are inherently complex and non-static.

Key Components of ParticleNet

1. EdgeConv Layers – Learning Dynamic Relationships

The core of **ParticleNet** is the **EdgeConv** layer, which replaces conventional graph convolution operations. In traditional **Graph Convolutional Networks (GCNs)**, nodes aggregate information from their fixed neighbors, but **EdgeConv dynamically recomputes neighborhood connections** in every layer.

Each EdgeConv layer consists of:

1. Edge Construction (k-Nearest Neighbors - KNN):
    - Each particle (node) is connected to its k-nearest neighbors based on Euclidean distance.
    - Unlike fixed adjacency matrices, these edges are recomputed at every layer.
2. Edge Feature Computation using MLPs:
    - Instead of directly operating on node features, EdgeConv computes edge features as:

$$e_{ij} = h_\theta(x_i, x_j - x_i)$$

where x_i and x_j are node feature vectors, and h_θ is a multi-layer perceptron (MLP).

    - The subtraction $x_j - x_i$ allows the network to capture relative positional information.
3. Feature Aggregation using Mean Pooling:
    - The computed edge features e_{ij} are aggregated using mean pooling across all neighbors.

- This helps capture higher-order relationships and makes the model robust to noise.

Thus, EdgeConv layers dynamically refine the graph structure, ensuring that the network learns meaningful local interactions between particles.

2. Multi-layer perceptrons (MLPs) for Feature Extraction

The MLP in EdgeConv is a fully connected neural network that acts as a feature transformation block.

- The MLP consists of multiple fully connected layers, followed by batch normalization and ReLU activation.
- It transforms the node and edge features into a high-dimensional space before aggregation.
- It allows the model to learn complex relationships beyond simple Euclidean distances.

Mathematically, the MLP in EdgeConv can be expressed as:

$$h\theta(xi,xj-xi)=ReLU(W2 \cdot \sigma(W1 \cdot (xi,xj-xi)))$$

where:

- W1and W2 are learnable weight matrices.
- σ is a non-linearity (e.g., ReLU activation function).
- Batch Normalization is applied to stabilize training.

3. Feature Aggregation – Capturing Global Relationships

After edge feature computation via MLPs, the aggregated features are pooled using mean aggregation:

$$xi'=1/k \sum_{j \in N(i)} eij$$

where:

- N(i) denotes the neighborhood of node i.
- xi′ is the updated feature vector for node i.

Unlike max-pooling (which only selects the most prominent feature), mean aggregation retains more detailed information about particle interactions.

4. Fully Connected (FC) Layers for Final Classification

Once the EdgeConv layers extract and refine the particle interactions, the global graph representation is computed using global mean pooling:

$$X_{graph} = 1/N \sum_{i=1}^{N} x_i$$

This graph-level representation is then fed into a series of fully connected layers:

Training & Evaluation

- The training loss function used is CrossEntropyLoss.
- The Adam optimizer is used with a learning rate of 0.001.
- The model is trained for 10 epochs.
- Evaluation Metrics:
    a. Accuracy
    b. ROC Curve (AUC)
    c. Confusion Matrix

<u>Results</u>

- The final validation accuracy is 0.7916 (79.16%).
- ROC AUC Score:
    - Gluon AUC = 0.87
    - Quark AUC = 0.87
- Confusion Matrix:
    - Gluon correctly classified: 2069, misclassified as quark: 401.
    - Quark correctly classified: 1889, misclassified as gluon: 641.

2nd Model: Graph Isomorphism Network (GIN)

Overview of GIN

The Graph Isomorphism Network (GIN) is a powerful GNN architecture specifically designed to distinguish graph structures effectively. Unlike EdgeConv (used in ParticleNet), which dynamically learns edge connections, GIN focuses on message-passing mechanisms using Multi-Layer Perceptrons (MLPs) and sum-based feature aggregation to capture complex relationships between nodes.

GIN was introduced by Xu et al. (2018) [2] to address the graph isomorphism problem, meaning it ensures that different graph structures do not collapse into the same representation—a common issue with traditional GCNs. This property makes it highly effective for particle physics applications, where fine-grained differences in jet structures are crucial for classification.

It consists of multiple GINConv layers, each using a Multi-Layer Perceptron (MLP) to aggregate node features from its neighbors through sum-based pooling. Unlike traditional Graph Convolutional Networks (GCNs) that use linear transformations, GIN employs learnable parameters (ϵ) to control self-loops, making it more expressive in capturing graph structures. The architecture includes three GINConv layers, each refining node embeddings before applying global add pooling to summarize the entire graph into a fixed-size representation. This pooled representation is passed through fully connected layers, which map it to classification logits for distinguishing between quark and gluon jets. The sum aggregation operation ensures that different graph structures produce distinct feature embeddings, making GIN highly expressive and well-suited for jet classification tasks.

Training and Performance

- Optimizer: Adam, learning rate = 0.001.
- Loss Function: Cross-Entropy Loss.
- Epochs: 20.
- Batch Size: 128

Results

- Validation Accuracy: 75.8%.
- ROC AUC Score:
    a. Gluon: 0.82
    b. Quark: 0.82
- Confusion Matrix:
    a. Gluon: 73% correctly classified, 27% misclassified.
    b. Quark: 78% correctly classified, 22% misclassified.

GIN performs slightly worse than ParticleNet (EdgeConv-based GNN), mainly because it operates on static KNN graphs, which do not dynamically adapt to particle interactions like EdgeConv.

Comparison of the Two Architectures

Architecture	Edge Type	Accuracy	AUC Score	Strengths	Weaknesses
ParticleNet (EdgeConv)	Dynamically learned edges	**79.16%**	**0.87**	Learns relationships dynamically, performs well on complex jet structures	Computationally expensive
GIN (Graph Isomorphism Network)	Fixed edges (KNN-based)	**75.8%**	**0.82**	Strong theoretical properties, effective on many GNN tasks	Less effective in learning jet interactions

Final Conclusion

- ParticleNet (EdgeConv) outperforms GIN in both accuracy (79.16%) and AUC (0.87).
- GIN still performs well, but fixed edges may limit its effectiveness in high-energy physics applications.
- Both models demonstrate that GNNs are powerful for jet classification, but choosing the right architecture is crucial for performance.

References

[1] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. "Dynamic Graph CNN for Learning on Point Clouds." *arXiv preprint arXiv:1801.07829* (2019). Link

[2] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. "How Powerful are Graph Neural Networks?" *arXiv preprint arXiv:1810.00826v3* (2019). Link