Quantum Rationale-Aware Graph Contrastive Learning for Jet Discrimination

Anonymous authors
Paper under double-blind review

Abstract

In high-energy physics, particle jet tagging plays a pivotal role in distinguishing quark from gluon jets using data from collider experiments. While graph-based deep learning methods have advanced this task beyond traditional feature-engineered approaches, the complex data structure and limited labeled samples present ongoing challenges. However, existing contrastive learning (CL) frameworks struggle to leverage rationale-aware augmentations effectively, often lacking supervision signals that guide the extraction of salient features and facing computational efficiency issues such as high parameter counts. In this study, we demonstrate that integrating a quantum rationale generator (QRG) within our proposed **Q**uantum **R**ationale-aware **G**raph **C**ontrastive **L**earning (QRGCL) framework significantly enhances jet discrimination performance, reducing reliance on labeled data and capturing discriminative features. Evaluated on the quark-gluon jet dataset, QRGCL achieves an AUC score of 77.53% while maintaining a compact architecture of only 45 QRG parameters, outperforming classical, quantum, and hybrid GCL and GNN benchmarks. These results highlight QRGCL's potential to advance jet tagging and other complex classification tasks in high-energy physics, where computational efficiency and feature extraction limitations persist. The source code for QRGCL is available at `https://anonymous.4open.science/r/QRGCL-1951`.

1 Introduction

Particle jet tagging, a fundamental task in high-energy physics, aims to identify the originating parton-level particles by analyzing collision byproducts at the Large Hadron Collider (LHC). While traditional approaches have relied on manually engineered features, modern deep learning methods offer promising alternatives for processing vast amounts of collision data (Kogler et al., 2019). The representation of jets as collections of constituent particles has emerged as a more natural and flexible approach compared to image-based methods, allowing for the incorporation of arbitrary particle features (Qu & Gouskos, 2020; Forestano et al., 2024). However, the challenge of limited labeled data in particle physics necessitates innovative solutions beyond purely supervised learning approaches. Self-supervised pretraining followed by supervised fine-tuning has shown particular promise in this domain. Self-supervised contrastive learning (CL) (Wang et al., 2025b;a; Xuan et al., 2024; Li et al., 2022) has gained significant attention in the field of graph neural networks (GNNs) (Veličković, 2023; Wu et al., 2021), leading to the development of graph CL (GCL). This approach involves pre-training a GNN on large datasets without manually curated annotations, facilitating effective fine-tuning for subsequent tasks (You et al., 2020).

A review of existing GCL approaches reveals a common framework that combines two primary modules: (1) graph augmentation, which generates diverse views of anchor graphs through techniques like node dropping, edge perturbation, and attribute masking, and (2) CL, which maximizes agreement between augmented views of the same anchor while minimizing agreement between different anchors. However, these methods face inherent challenges due to the complexity of graph structures, where random augmentations may obscure critical features, potentially misguiding the CL process. In response to these challenges, recent studies have

*Corresponding Author(s)

shifted focus towards understanding the invariance properties (Misra & van der Maaten, 2020; Dangovski et al., 2022) of GCL. The necessity for augmentations was emphasized to maintain semantic integrity, arguing that high-performing GCL frameworks should enhance instance discrimination without compromising the intrinsic semantics of the anchor graphs. Building on this foundation, invariant rationale discovery (IRD) techniques (Li et al., 2022; Wu et al., 2022; Chang et al., 2020) were proposed, aligning closely with the objectives of GCL. These techniques highlight the importance of identifying critical features that inform predictions effectively.

Despite these advancements, gaps remain in effectively leveraging rationales for augmentation. Existing frameworks often lack supervision signals, hindering their ability to reveal and utilize the most salient features effectively. As the amount of available data increases and the computational cost of deep learning networks rises, significant computing resources will be required to execute these algorithms efficiently (LeCun et al., 2015). To handle computational complexity, we see a move from classical networks that process bits to quantum networks that use qubits (Havlícek et al., 2019; Jahin et al., 2023). Quantum networks leverage unique properties such as superposition and entanglement, allowing them to represent 2^n characteristics from n two-dimensional complex vectors. In contrast to classical networks, which scale linearly in expressiveness, quantum networks show exponential growth as the sample size n increases (Larocca et al., 2023; Yoshioka et al., 2024). Recognizing the pivotal role of the rationale generator in the GCL framework, we propose enhancing this component with a quantum-based subroutine to evaluate its potential for performance improvement. Our proposed **Q**uantum **R**ationale-aware **G**raph **C**ontrastive **L**earning (QRGCL) addresses these limitations by integrating a quantum rationale generator (QRG) that autonomously identifies and reveals salient features within the graphs, ensuring that the generated augmentations retain their discriminative power. Our approach incorporates the state-of-the-art ParticleNet (Qu & Gouskos, 2020) GNN encoder network, followed by a projection head, culminating in a multi-layer perceptron (MLP) classifier. Experimental results on the quark-gluon jet tagging dataset showcase the superiority of our approach, proving its capacity to capture distinguishing semantic nodes and significantly outperforming current state-of-the-art GCL and GNN methods (classical, quantum, and hybrid).

Our **main contributions** are: **(i)** We propose a novel hybrid quantum-classical framework, Quantum Rationale-aware Graph Contrastive Learning (QRGCL), that integrates a quantum rationale generator to identify salient substructures in graph-structured particle physics data for improved CL; **(ii)** We design a parameter-efficient QRG based on a 7-qubit variational quantum circuit, enabling salient feature extraction with only 45 trainable parameters; **(iii)** We introduce a new quantum-enhanced contrastive loss that incorporates rationale-aware, contrastive pairs, and alignment losses, with quantum fidelity as a distance metric; **(iv)** We conduct extensive experiments on the quark-gluon jet tagging dataset, showing that QRGCL significantly outperforms classical, quantum, and hybrid benchmarks in terms of AUC, while maintaining a compact and computationally efficient architecture.

2 Background

2.1 Contrastive Representation Learning

Contrastive representation learning (Peng et al., 2025; Mo et al., 2024; Kottahachchi Kankanamge Don et al., 2024; Don & Khalil, 2024; Padha & Sahoo, 2024) is an effective framework for extracting meaningful representations from high-dimensional data by mapping it into a lower-dimensional space. CL uses a self-supervised approach to differentiate between positive pairs (similar data) and negative pairs (dissimilar data), particularly when labeled data is scarce. The framework consists of three components:

1. **Data Augmentation Module:** This module generates multiple invariant views of a given sample using invariance-preserving transformations. The goal is to create variations of the same input that retain essential characteristics, forming positive pairs, while views from different samples are negative pairs. For instance, augmenting a quark jet should retain its distinguishing features even when transformations like noise addition or spatial shifts are applied. This ensures that the learned representations remain general and robust to different data variations.

2. **Encoder Network:** The augmented views are processed through an encoder, which maps each view into a lower-dimensional embedding space. Each view in a pair is passed independently through the encoder, generating embeddings that capture the intrinsic features of the input data while abstracting away specific details.

3. **Projection Network:** While optional, the projection network is often used to adjust the dimensionality of the embeddings, enabling fine-tuning of the representation space. Typically, this is implemented as a single linear layer that transforms the encoded embeddings into a space suitable for CL objectives.

The goal is to train the encoder to minimize the distance between positive pair embeddings and maximize the distance between negative pair embeddings, enhancing representation quality for downstream tasks like classification.

2.2 Quantum Contrastive Learning (QCL)

Recent studies have begun to explore how quantum computing can advance traditional CL frameworks (Jia et al., 2025). Quantum systems can potentially offer superior computational capabilities, allowing for more complex feature extraction and representation learning. One study (Jaderberg et al., 2022) proposed a hybrid quantum-classical model for self-supervised learning, showing that small-scale QNNs could effectively enhance visual representation learning. By training quantum and classical networks together to align augmented image views, they achieved higher test accuracy in image classification than a classical model alone, even with limited quantum sampling. Another approach (Kottahachchi Kankanamge Don et al., 2024) integrates Supervised CL (SCL) with Variational Quantum Circuits (VQC) and incorporates Principal Component Analysis (PCA) for effective dimensionality reduction. This method addresses the limitations posed by scarce training data and showcases potential in medical image analysis. Experimental results reveal that this model achieves impressive accuracy across various medical imaging datasets, particularly with a minimal number of qubits (2 qubits), underscoring the benefits of quantum computing. A different research effort presents Q-SupCon (Don & Khalil, 2024), a fully quantum-enhanced Supervised CL (SCL) model tailored to tackle issues related to data scarcity. Experiments demonstrate that this model yields significant accuracy in image classification tasks, even with very limited labeled datasets. Its robust performance on actual quantum devices illustrates its adaptability in scenarios with constrained data availability. Furthermore, a quantum-enhanced self-supervised CL framework has been proposed for effective mental health monitoring (Padha & Sahoo, 2024). This framework leverages a quantum-enhanced Long Short-Term Memory (LSTM) encoder to enhance representation learning for time series data through CL. The results indicate that this model significantly outperforms traditional self-supervised learning approaches, achieving high F1 scores across multiple datasets. To take advantage of QCL, as evident in these studies, we attempted to replace the classical rationale generator with a VQC in our proposed framework.

2.3 Rationale-Aware Graph Contrastive Learning (RGCL) Concept

RGCL (Li et al., 2022) represents a self-supervised CL approach that overcomes several limitations common to traditional graph CL (GCL) frameworks. Standard GCL methods often suffer from challenges such as augmentation strategies that can inadvertently alter or remove critical graph structure and semantics, and attempts to preserve graph-specific domain knowledge sometimes result in overfitting, limiting the model's adaptability to diverse, unseen data (Ji et al., 2024). RGCL addresses these issues by focusing on the concept of *rationale learning*, where the essential, discriminative information for graph classification is typically concentrated within a subset of nodes or edges in the graph. In RGCL, this discriminative subset, or *rationale*, is identified and emphasized during training, allowing the model to prioritize meaningful patterns while minimizing reliance on irrelevant features. In RGCL, specialized neural networks, known as the *rationale generator (RG)*, are used to assign importance scores to each node. This generator evaluates each node's contribution to the graph's overall representation. The higher-score nodes form the rationale subset, while the remaining nodes comprise the *complement* subset. The rationale subset undergoes targeted augmentations during training, capturing the core discriminative features essential for downstream tasks. In contrast, the complement subset is augmented to encourage the exploration of less critical correlations, thus

avoiding overfitting and improving generalization. RGCL pipeline leverages these dual views—*rationale* and *complement*—to guide the encoder network in learning a balanced feature space. By focusing on rationale views, the model learns robust, task-relevant features, while the complement views prevent it from becoming overly sensitive to spurious relationships, fostering a more generalized understanding.

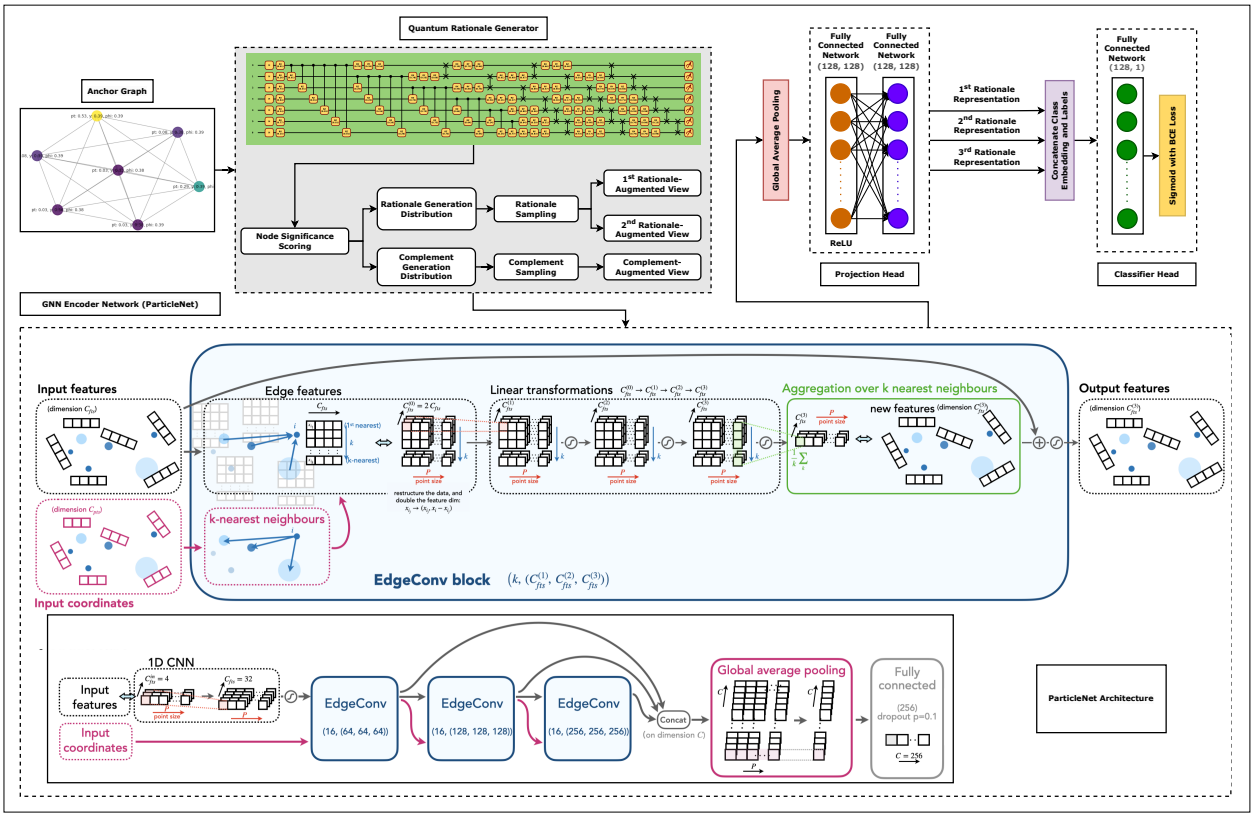


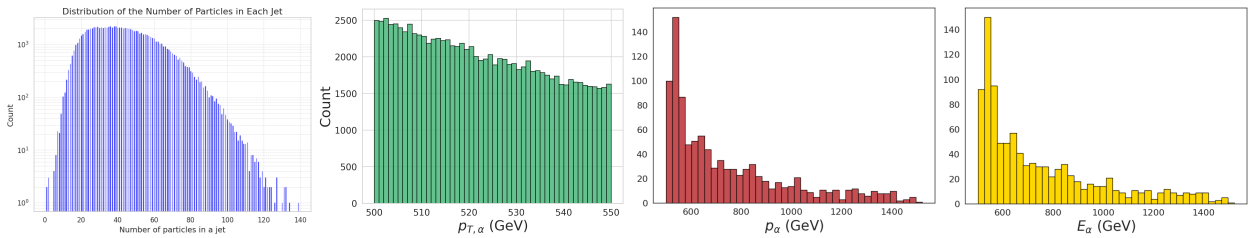
Figure 1: QRGCL framework. The quantum rationale generator identifies a discriminative subset of nodes in the original graph. The rationale generator shared GNN-encoder and projection head is jointly optimized by minimizing the combined loss.

3 Methodology

3.1 Dataset and Preprocessing

3.1.1 High-Energy Physics Dataset

This study uses the *Pythia8 Quark and Gluon Jets for Energy Flow* (Komiske et al., 2019) dataset, a well-established dataset in high-energy physics. The dataset comprises two million simulated particle jets, evenly divided between one million quark-originated jets and one million gluon-originated jets. These jets are generated through collision events at the Large Hadron Collider (LHC) with a center-of-mass energy $\sqrt{s} = 14\,\text{TeV}$. Jets were selected based on their transverse momentum range p_T^{jet} between 500 and 550 GeV and their pseudorapidity $|\eta^{\text{jet}}| < 1.7$. Each jet α is labeled as a quark jet ($y_\alpha = 1$) or a gluon jet ($y_\alpha = 0$), providing a binary classification target for model training. The kinematic distributions of the jets, along with the particle count in each jet, are visualized in Figure 2, highlighting the differences between the quark and gluon jet populations. Each particle i within a jet is characterized by several key attributes: transverse momentum $p_{T,\alpha}^{(i)}$, rapidity $\eta_\alpha^{(i)}$, azimuthal angle $\psi_\alpha^{(i)}$, and its Particle Data Group (PDG) identifier $I_\alpha^{(i)}$.



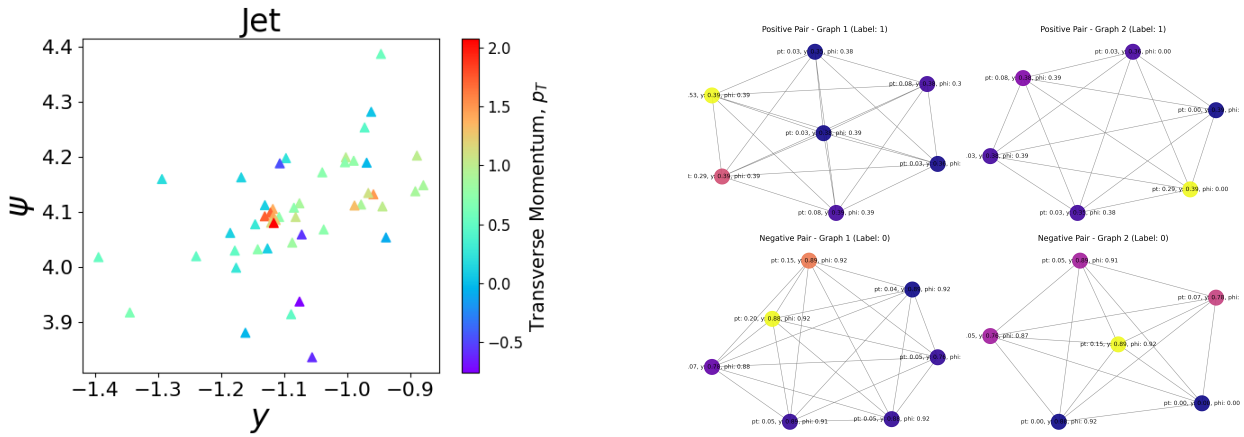
(a) (b) (c) (d)

Figure 2: Distribution of (a) number of particles in each jet, (b) transverse momentum, (c) total momenta, and (d) energy.

3.1.2 Graph Representation of Jets

A graph G is defined as a set of nodes V and edges E, represented as $G = \{V, E\}$. Each node $v^{(i)} \in V$ is connected to its neighboring nodes $v^{(j)}$ through edges $e^{(ij)} \in E$. In the context of this study, each jet α is modeled as a graph J_α, where the nodes $v_\alpha^{(i)}$ represent the particles in the jet, and the edges $e_\alpha^{(ij)}$ represent the interactions between these particles. Each node $v_\alpha^{(i)}$ is associated with a set of features $h_\alpha^{(i)}$, which describe its properties, while the edges have attributes $a_\alpha^{(ij)}$ that characterize the relationship between connected nodes.

The number of nodes in each graph can vary significantly, reflecting the varying number of particles within each jet. This variability is particularly pronounced in particle physics, where jets can differ greatly in their particle multiplicity. Consequently, each jet graph J_α is composed of m_α particles, each with l distinct features that capture various physical properties. This graph representation provides a natural way to encode the complex interactions within jets, enabling models to leverage the relational structure among particles. An illustration of this graph-based data representation, along with an example jet depicted in the (ψ, y) plane, is shown in Figure 3a. Here, particles are visualized as nodes and their interactions as edges, offering a clear view of the underlying structure and relationships within the jet.



(a) (b)

Figure 3: Plot of (a) a sample jet shown in (ψ, y) plane with each particle color-coded by its $p_{T,\alpha}^{(i)}$, (b) a sample of graph views used in our CL-based approach. Each graph represents a jet as a collection of nodes (particles) with associated physics-based features. The graphs are undirected, reflecting the bidirectional nature of interactions between particles.

3.1.3 Feature Engineering

Additional kinematic variables are derived from the original features $(p_{T,\alpha}^{(i)}, y_\alpha^{(i)}, \psi_\alpha^{(i)})$ using the 'Particle' package to enhance the model's ability to learn from the data. These engineered features include transverse mass, energy, and Cartesian momentum components, providing a more complete description of each particle's dynamics. More details about these engineered features are shown in Appendix A.1.

3.1.4 Edge Construction and Attributes

These features are then normalized by their maximum values across all jets to ensure consistent input scales, enhancing the stability of the training process. Edges between particles in a jet are defined based on the spatial proximity of particles in the (ψ, y) plane, calculated using the Euclidean distance:

$$\Delta R_\alpha^{(ij)} = a_\alpha^{(ij)} = \sqrt{\left(\psi_\alpha^{(i)} - \psi_\alpha^{(j)}\right)^2 + \left(y_\alpha^{(i)} - y_\alpha^{(j)}\right)^2} \tag{1}$$

This metric measures the angular separation between two particles, capturing their spatial relationships within the jet. The resulting edge attribute matrix $\Delta R_\alpha^{(ij)} = a_\alpha^{(ij)}$ is used as input for graph-based models to encode the spatial structure of the jet.

3.1.5 Graph-Based Augmentation and Contrastive Learning

We applied graph-based CL techniques to enhance the discriminative capability of our models by generating augmented graph views. The augmentation strategies included node dropping, edge perturbation, feature masking, and jet-specific transformations. Details of the augmentation pipeline, view pairing, and the construction of positive and negative pairs are provided in Appendix A.2.

3.1.6 Enforcing Infrared and Collinear Safety

To ensure compliance with infrared and collinear (IRC) safety, we adopt perturbation and regularization techniques inspired by the principles of QCD, as outlined by Dillon et al. (Dillon et al., 2022). Further theoretical and implementation details are provided in Appendix A.2.1.

3.1.7 Data Splitting

To ensure manageable computational complexity and adapt the model for quantum processing, we focused on jets containing at least ten particles, resulting in a dataset of $N = 1,997,445$ jets, of which $997,805$ were classified as quark jets. Unlike classical graph neural networks (GNNs), which offer flexibility in adjusting the number of hidden features, quantum networks are constrained by the scaling of quantum states and Hamiltonians. Specifically, the quantum computational cost scales as 2^n, where n represents the number of qubits, corresponding to the number of nodes (n_α) in the graph. Each node represents a particle in the jet, making jets with many particles challenging to handle due to the exponential growth in quantum computational requirements.

To address the challenge of varying particle numbers in jets, we simplified the problem by limiting the number of active nodes (particles) per jet to $n_\alpha = 7$. This was done by selecting the seven particles with the highest transverse momentum (p_T) from each jet. Consequently, each jet graph is represented by a feature set $h_\alpha = (h_\alpha^{(1)}, h_\alpha^{(2)}, \ldots, h_\alpha^{(7)})$, where each $h_\alpha^{(i)} \in \mathbb{R}^8$ corresponds to the enriched feature vector of a particle. The complete representation of each jet is thus given by $h_\alpha \in \mathbb{R}^{7\times 8}$, capturing key physical attributes of the selected particles. A subset of $N = 12,500$ jets was randomly selected for model training, with 10,000 jets used for training, 1,250 for validation, and 1,250 for testing. These subsets maintained the original class distribution, resulting in 4,982 quark jets in the training set, 658 in the validation, and 583 in the testing set.

3.2 Proposed QRGCL

Quantum rationale-aware GCL (QRGCL) consists of 4 major components, as illustrated by Figure 1: rationale generator (RG), encoder network, projection head, and loss function.

3.2.1 Quantum Rationale Generator (QRG)

The QRGCL model substitutes its classical RG (CRG) (Li et al., 2022) with a quantum RG (QRG). This component is crucial in generating augmented graph representations by assigning significance scores to each node. The QRG is built using a 7-qubit parameterized quantum circuit (PQC), where each qubit represents a node in the graph.

The quantum circuit for the QRG consists of 3 main components, as shown in Figure 4: *data encoding*, *parameterized unitaries*, and *entanglement*. The encoding process starts by initializing each qubit, typically using a Hadamard (H) gate to create a uniform superposition state:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \tag{2}$$

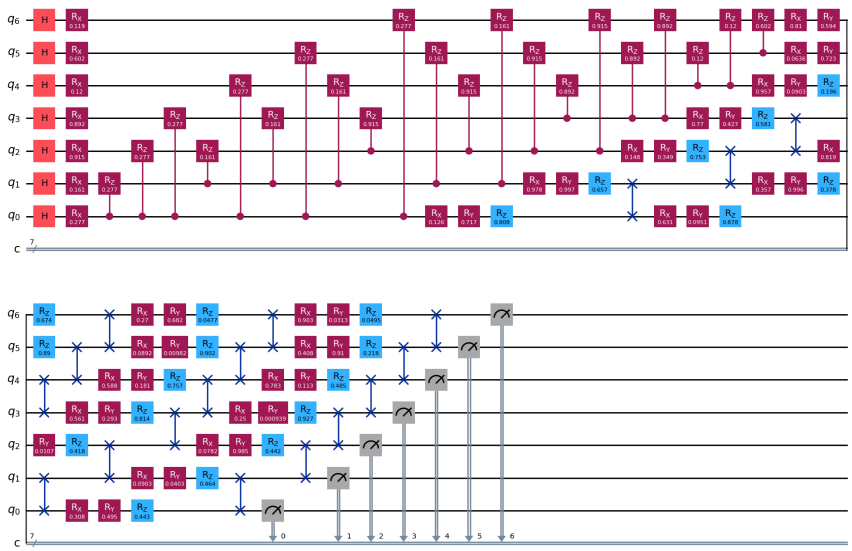


Figure 4: QRG circuit of the proposed QRGCL. The circuit operates on seven qubits, with each qubit corresponding to a node in the graph. The top portion shows the data encoding stage, where each qubit is initialized using an H gate followed by RX-based angle encoding node features. CRZ gates encode edge relationships between qubits. The bottom portion includes parameterized rotations (RX, RY, RZ) for adaptable representations and entanglement layers using $SWAP$ gates. Measurement results are obtained on a computational basis, with classical registers collecting the output.

Next, node feature vectors are encoded using rotation gates, with various encoding options such as RX, RY, RZ, or H gates. The specific choice of encoding can be customized, with RX encoding being implemented for proposed angle-based representations: Node feature vectors $\mathbf{x}_i$ are encoded as rotation angles using RX, RY, or RZ gates. For example, the RX gate is defined as:

$$RX(\theta) = \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \tag{3}$$

where θ is determined by the value of $\mathbf{x}_i$. Edge relationships are encoded using controlled-phase (CRZ) gates between pairs of qubits. This is a diagonal, asymmetric gate that applies a phase to the target qubit depending on the control qubit's state:

$$CRZ(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix} \tag{4}$$

After encoding, each qubit undergoes parameterized $U3$ gates, defined as:

$$U3(\theta, \phi, \lambda) = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i(\phi+\lambda)}\cos(\theta/2) \end{pmatrix} \tag{5}$$

which allows the QRG to learn adaptable representations through trainable parameters θ, ϕ, and λ. These parameterized gates form the trainable part of the circuit, allowing the QRG to adaptively learn the significance scores based on the input data during training. The parameters of these gates are optimized through gradient-based approaches. Entanglement is introduced using a flexible choice of entanglement layers, such as $CNOT$, CZ, $SWAP$, or their butterfly-pattern variants. These entanglement patterns are designed based on the graph's structure and ensure that important correlations between nodes are captured. For example, $SWAP$ gates can exchange quantum states between qubits, preserving relationships between specific nodes:

$$SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

The output of the QRG is obtained by measuring the qubits on a computational basis. The squared amplitudes of states with a Hamming weight of 1 (e.g., $|00001\rangle, |00010\rangle, \ldots$) provide node significance scores normalized into a discrete probability distribution. With the QRG generating significance scores, augmented views are created for the downstream CL process. For an optimistic view, nodes with the highest significance scores are retained, preserving the most relevant information for classification. Negative views, on the other hand, are constructed by using less significant nodes or by introducing random variations, helping the model learn to distinguish between genuinely informative structures and noise.

Integration into the encoder. In practice, the QRG's measured probabilities are used to select the top-k most salient nodes (and their incident edges) to form a rationale subgraph. This subgraph is then fed to the classical ParticleNet encoder, which applies EdgeConv on the selected nodes to produce embeddings for CL and, later, classification. This makes explicit the data flow: full jet $\rightarrow$ QRG importance scores $\rightarrow$ selected subgraph $\rightarrow$ ParticleNet $\rightarrow$ projection/classifier.

3.2.2 Encoder Network

We used the ParticleNet (Qu & Gouskos, 2020) model as our encoder to convert the augmented views of input particle features into low-dimensional embeddings. ParticleNet is a graph-based neural network optimized for jet tagging, leveraging dynamic graph convolutional neural networks (DGCNN) to process unordered sets of particles, treating jets as particle clouds. More details about ParticleNet are shown in Appendix B.1.

3.2.3 Projection Head

Our architecture incorporates a projection head that effectively maps high-dimensional embeddings from the encoder into a compact, CL-optimized latent space. This distinction allows the encoder to learn features suited for downstream tasks while the projection head focuses on maximizing contrastive effectiveness. This projection head employs an initial linear transformation to reduce the dimensionality of the input embedding (128-dimensional) to a latent space of the same dimensionality, followed by a ReLU activation and a final linear layer. This layered design allows the projection head to align representations tightly with the CL objective by enforcing mutual information between anchor and rationale representations in the latent space.

3.2.4 Quantum-Enhanced Contrastive Loss

QRGCL model uses a carefully designed loss function that integrates multiple elements: InfoNCE, alignment, uniformity, rationale-aware loss (RA loss), and contrastive pair loss (CP loss), to optimize the learning of quantum-enhanced embeddings. More details about these losses are shown in Appendix B.2.

3.2.5 Classifier Head

The classifier head follows our GCL framework and consists of a simple neural network with a 128-neuron single linear layer and is designed to perform binary jet discrimination, mapping features to a single output with a sigmoid activation for probabilistic classification.

3.3 Benchmark Models

We developed two classical models (GNN and EQGNN), three quantum models (QGNN, EQGNN, and QCL), five hybrid classical-quantum models (CQCL, 3 QCGCL variants, and QRGCL with $RX + H$ encoding) (see Table 3), and the classical counterpart of QRGCL, i.e., CRGCL (see Table 2). More details are shown in Appendix C.

Table 1: Hyperparameter optimization results of the proposed QRGCL. **Bold** indicates the best performance.

Parameter type	Parameter	Test Accuracy (↑)	AUC (↑)	F1 Score (↑)
Encoder type	Amplitude	69.30%	74.69%	69.30%
	IQP	68.90%	76.10%	68.47%
	Displacement-amplitude	66.17%	72.10%	65.97%
	Displacement-phase	70.17%	76.22%	69.13%
	RY	67.00%	73.93%	67.00%
	RX	**70.80%**	**76.30%**	**69.60%**
	RZ	62.90%	67.32%	62.80%
	H	**70.83%**	**76.71%**	**69.65%**
	Phase	68.50%	74.21%	68.30%
Learning rate	**1E-03**	**70.30%**	**76.20%**	**70.29%**
	1E-04	68.60%	74.41%	68.39%
	1E-02	69.50%	75.81%	69.47%
	3E-03	63.90%	68.83%	63.70%
	5E-03	66.00%	70.89%	65.99%
Entanglement type	CNOT	69.80%	76.58%	69.79%
	CZ	68.20%	73.45%	68.14%
	SWAP	**71.00%**	**76.20%**	**70.79%**
	CNOT Butterfly	68.00%	74.63%	67.98%
	CZ Butterfly	67.70%	73.20%	67.69%
	SWAP Butterfly	68.00%	73.21%	67.79%

Table 2: Comparison between classical and quantum RG of RGCL. **Bold** indicates the best performance.

Parameter type	Parameter	Classical RG			Quantum RG		
		Test Accuracy (↑)	AUC (↑)	F1-score (↑)	Test Accuracy (↑)	AUC (↑)	F1-score (↑)
Nodes per graph	**7**	70.40%	**76.31%**	69.13%	70.80%	**75.91%**	69.86%
	8	68.20%	74.09%	67.90%	70.80%	75.61%	70.50%
	9	68.40%	73.38%	67.15%	67.60%	74.46%	67.54%
	10	67.00%	73.37%	66.50%	70.70%	75.74%	70.30%
Number of layers	2	68.80%	74.51%	68.80%	66.80%	71.42%	66.49%
	3	66.70%	73.58%	66.70%	68.60%	**74.71%**	68.50%
	4	71.20%	**76.29%**	71.08%	66.40%	72.34%	66.39%
	5	63.00%	67.45%	62.99%	67.40%	71.16%	67.28%
Augmentation ratio	**0.1**	67.80%	**74.09%**	67.78%	72.10%	**78.78%**	72.10%
	0.2	64.60%	69.78%	64.49%	65.60%	71.14%	65.59%
	0.3	63.70%	69.57%	63.70%	71.30%	76.95%	71.28%

Table 3: Performance benchmarking and ablation test of the proposed QRGCL. **Bold** indicates the best performance.

Model	Test accuracy (↑)	AUC (↑)	F1 score (↑)	# Parameters (↓)	n_α (↓)	n_{layer} (↓)	Batch size	Epoch	Encoder
QGNN	72.20%±2.62%	70.37%±2.09%	72.09%±2.44%	5156	3	6	1	19	MLP + H
GNN	73.90%±1.78%	63.36%±0.89%	73.93%±1.34%	5122	3	5	64	19	MLP
EGNN	73.90%±0.89%	67.88%±0.45%	73.55%±0.89%	5252	3	4	64	19	MLP
EQGNN	71.40%±2.23%	74.36%±1.78%	71.22%±1.45%	5140	3	6	1	19	MLP + H
QCL	50.40%±1.29%	53.26%±0.48%	51.49%±0.68%	280	3	3	256	1000	Amplitude
CQCL	50.00%±0.75%	49.79%±1.50%	48.33%±1.29%	250	3	3	256	1000	Amplitude
QCGCL	57.44%±1.46%	62.26%±1.56%	56.72%±1.56%	7448	7	6	128	50	Angle (RY+RX)
QCGCL	65.39%±1.02%	71.07%±0.48%	63.80%±1.02%	7448	7	6	128	50	Angle (RY)
QCGCL	65.33%±1.26%	70.83%±0.78%	64.52%±1.53%	7448	7	6	128	50	Amplitude + Angle (RY)
QRGCL variant	70.93%±0.99%	77.30%±1.77%	70.40%±1.67%	126015	7	3	2000	50	RX + H
Proposed QRGCL	71.50%±0.82%	**77.53%±0.88%**	70.35%±0.82%	126015	7	3	2000	50	**H + RX**

Clarification: Models marked "CL" used CL but did not use a rationale generator. All other models (EGNN, EQGNN, GNN, QGNN) use full subgraphs of size n_α without rationale selection or augmentation.

Table 4: Scalability comparison of classical, quantum, and hybrid models as the number of active nodes per graph (n_α) increases beyond 3. Entries marked "–" indicate training failures due to out-of-memory (OOM) errors resulting from exponential circuit width growth at $n_\alpha = 7$.

Model	Test Accuracy	AUC	F1-score	n_α	#Parameters	n_{layer}	Batch size	Epoch	Encoder
EGNN	54.2%	64.39%	68.52%	7	5252	5	64	19	MLP
EQGNN	47.8%	55.13%	30.92%	7	990100	6	1	19	MLP + H
QGNN	54.6%	43.90%	54.62%	7	1021076	6	1	19	MLP + H
GNN	52.2%	57.76%	35.81%	7	5252	4	64	19	MLP
QCL	44.8%	48.28%	61.88%	5	384	3	256	1000	Amplitude
QCL	–	–	–	7	–	3	256	1000	Amplitude
CQCL	45.2%	48.42%	60.74%	5	354	3	128	50	Angle (RY+RX)
CQCL	–	–	–	7	–	3	128	50	Angle (RY)

4 Experimental Setup

4.1 Simulation Tools and Environment

We implemented all the models using the *PyTorch 2.2.0* (Paszke et al., 2019) framework for classical computations and *Pennylane 0.38.0* (Bergholm et al., 2018) and *TorchQuantum 0.1.8* (Wang et al., 2022) for quantum circuit simulation. We used the *Deep Graph Library (DGL) 2.1.0+cu121* (Zheng et al., 2020) to handle graph operations and the *Qiskit 0.46.0* (Javadi-Abhari et al., 2024) framework to simulate quantum circuits. The computing infrastructure consisted of Intel(R) Xeon(R) CPUs (x86) with a clock frequency of 2 GHz, equipped with 4 vCPU cores and 30 GB of DDR4 RAM. For GPU acceleration, we leveraged two NVIDIA T4 GPUs, each with 2560 CUDA cores and 16 GB of VRAM, significantly boosting the performance of deep learning tasks.

4.2 Hyperparameters and Configurations

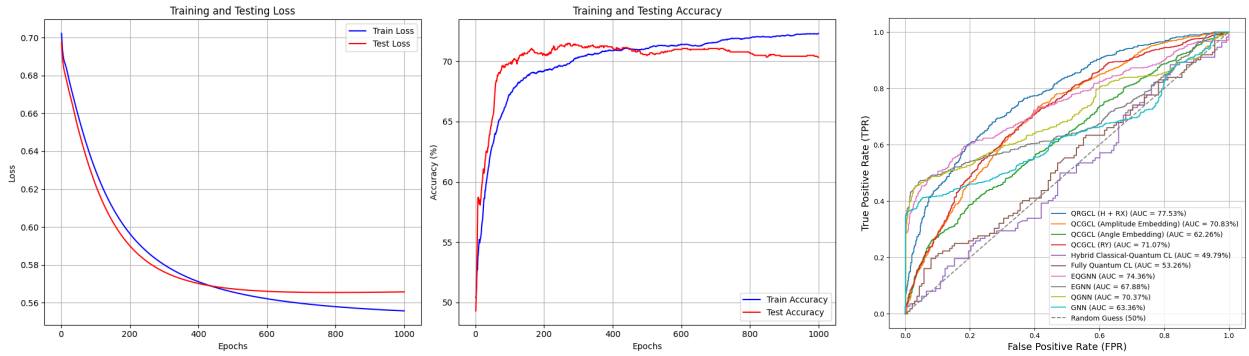
We varied the hyperparameters related to CRG, QRG, data augmentation, and training for the proposed QRGCL model, highlighting the test accuracy, AUC, and F1 scores across various encoder types, learning rates (LR), and entanglement strategies. We used Adam optimizer with a 1×10^{-3} learning rate across all the models and a binary cross-entropy loss function. 10-fold cross-validation was performed for each model, and the mean and standard deviation were calculated for each metric. The hidden feature size for classical GNN and EGNN was 10, while for the QGNN and EQGNN, it was 2^{n_α}=8. In the case of QCL and CQCL, n_{layer} stands for the depth of the quantum circuit, and in other models, it refers to the number of GNN layers. In Table 2, each row varies a single hyperparameter while keeping all others fixed to a shared default configuration ($n_\alpha = 7$ nodes, two GNN layers, 10% dropout, GAT encoder, embedding dimension 128, hidden size 256, output dimension 128, no weight decay, augmentation ratio 0.1, temperature 0.1, batch size 2000, and for QRGCL only: 3 quantum layers). Specifically, the third row sets the number of GNN layers to 4 for CRGCL while all other parameters remain at their defaults. No joint (pairwise) hyperparameter optimization was performed due to computational constraints; thus, comparisons reflect controlled one-factor-at-a-time sweeps under consistent training settings.

The choice of epochs and batch sizes varies based on the computational requirements of classical and quantum models. Classical models like GNN and EGNN use larger batch sizes (64) and fewer epochs (19) to leverage efficient gradient updates, allowing faster convergence. In contrast, quantum models such as QGNN and EQGNN use much smaller batch sizes (1) due to quantum hardware limitations, yet still require 19 epochs to ensure adequate learning despite more minor updates per step. For fully quantum and hybrid models like QRGCL and QCGCL, larger batch sizes (2000) and more epochs (50) are used due to the longer backpropagation caused by the complex custom loss function, with multiple loss components ensuring that the model has sufficient time to learn robust quantum-based representations, as the slower convergence can hinder effective learning.

Training Protocol. For fair comparison, classical GNN, EGNN, and EQGNN baselines were trained using fully supervised learning, whereas QCL, CQCL, and QCGCL adopted contrastive objectives. Only QRGCL and its classical counterpart CRGCL employed rationale-aware contrastive pretraining followed by fine-tuning, isolating the contribution of the QRG. We adopt a two-stage procedure for QRGCL: (i) self-supervised pretraining for 50 epochs using a rationale-aware contrastive objective $\mathcal{L}_{\text{RA}} + \lambda\,\mathcal{L}_{\text{CP}} + \alpha\,\mathcal{L}_{\text{align}} + \beta\,\mathcal{L}_{\text{uniform}} + \delta\,\mathcal{L}_{\text{InfoNCE}}$, and (ii) supervised fine-tuning with a linear classifier for 1000 epochs on the learned graph-level embeddings (the encoder is frozen). This step evaluates the quality of learned representations in a supervised downstream setting.

5 Results and Discussion

We evaluated the performance of QRGCL against two classical models (GNN and EQGNN), three quantum models (QGNN, EQGNN, and QCL), and five hybrid classical-quantum models (CQCL, three variants of QCGCL, and QRGCL with RX+H encoding). The area under the curve (AUC) was selected as the benchmark metric due to its effectiveness in assessing binary classification performance across all thresholds. The number of trainable parameters of CRG was 1,073, compared to 45 for QRG.



(a) (b)

Figure 5: (a) Training and testing dynamics of QRGCL over 1000 epochs. The left graph illustrates loss curves, while the right graph presents accuracy curves. (b) ROC-AUC curve for all the benchmarked models

The hyperparameters specific to QRGCL include encoder type, learning rate, and entanglement type, as detailed in Table 1. Among the encoders, RX and H achieved the highest AUC values at 76.30% and 76.71%, respectively, while displacement-amplitude and RZ encodings performed poorly, with AUCs of 72.10% and 67.32%. The optimal learning rate of 1×10^{-3} produced the highest AUC of 76.20%, with higher rates resulting in decreased performance. The $SWAP$ entanglement type yielded the best overall results, achieving an AUC of 76.20%. $CNOT$ and CZ entanglements performed strongly, while other configurations underperformed. For the proposed QGRCL, two possible combinations of RX and H encoders, the learning rate of 1×10^{-3} and $SWAP$ entangler were tried.

For both the classical RGCL and QRGCL models, tunable parameters included the number of nodes per head (n_α) ranging from 7 to 10, the number of classical or quantum layers (n_{layer}) ranging from 2 to 5, and the augmentation ratio ranging from 0.1 to 0.3. As shown in Table 2, QRGCL outperforms the classical RGCL with 8 and 10 nodes, achieving higher AUC values. At 3 layers, QRGCL surpasses the classical model, achieving 74.71% compared to 73.58%. However, it underperformed at 2 and 4 layers. With a 0.1 augmentation ratio, QRGCL significantly outperformed the classical approach, achieving 78.78% compared to 74.09%. The optimal parameters for QRGCL were found to be $n_\alpha = 7$, $n_{layer} = 3$, and an augmentation ratio of 10%.

Based on Table 3 and Figure 5b, it is evident that our proposed QRGCL model achieves the highest AUC score (77.53%). We took the best two well-performing encoders from Table 1 and developed two variants by hybridizing them, and further experimentations revealed that H followed by RX gate is more robust. Additionally, the relatively large number of parameters in the QRGCL is due to the utilization of the ParticleNet GNN encoder, which possesses 125k parameters, while the QRG circuit only has 45 parameters. Tables 1, 2, and 3 complement the ablation studies for QRGCL by presenting results for configurations without rationale-awareness (QGNN, GNN, EGNN, EQGNN), analyzing VQC components (such as encoding variants, entanglement structures, and variations in qubits and layers), exploring different classical-quantum interfaces (including quantum-only and hybrid architectures, as well as a classical-only baseline), and examining the effects of rationale-guided data augmentation. Figure 5a shows that both training and testing losses decrease steadily in QRGCL, indicating practical training with a continuous reduction in error. By around 800 epochs, the losses stabilize, suggesting that the model has reached a convergence point where further training yields minimal improvements. The small gap between the two loss curves suggests minimal overfitting and generalization of test data. Both training and testing accuracies rise sharply within the first 200 epochs, reflecting rapid learning and an effective initial adjustment of model parameters. Around 800 epochs, both accuracies stabilize. The test accuracy levels off slightly above 70%, while training accuracy remains close but slightly lower. This stability indicates that QRGCL has reached its maximum performance capacity on this dataset, with limited fluctuations.

Scalability. Models ending with "CL" in Table 4 used contrastive learning but lacked rationale generation. For other models, increasing n_α corresponds to including more raw particle features in the encoder without explicitly learning which ones are most informative. As n_α grows, quantum and hybrid models quickly encounter exponential scaling in circuit width and memory usage, leading to failed runs for QCL and CQCL at $n_\alpha = 7$. In contrast, classical graph-based models such as GNN and EGNN maintain stable accuracy and F1-scores, reflecting linear computational scaling. QGNN and EQGNN remain trainable but incur a steep parameter cost ($\approx 10^6$) without proportional performance gains. The absence of consistent improvement, and occasional degradation, with larger n_α supports our central finding: simply enlarging the particle subgraph does not enhance jet discrimination unless the most salient constituents are identified, as done by QRGCL. These results express that rationale-aware selection, rather than graph size, drives scalability and efficiency in quantum graph learning.

6 Broader Impacts

QRGCL enables efficient, low-supervision feature extraction that can accelerate particle physics discoveries, inspire quantum-augmented ML methods, raise ethical and interpretability considerations, and generalize to broader graph-structured problems across science and industry.

7 Conclusion

This paper introduced QRGCL, a hybrid quantum-classical rationale-aware GNN for quark-gluon jet classification, integrating quantum computation with GCL. Our results show that QRGCL outperforms two classical, three quantum, and five hybrid classical-quantum models, achieving an AUC of 77.53%. Hyperparameter tuning indicated that the $H + RX$ encoder and the $SWAP$ entanglement gate optimized performance, emphasizing the importance of careful circuit design in quantum models. QRGCL performed best on smaller graphs (7 nodes) with low augmentation (0.1), achieving an AUC of 78.78%, indicating their value for compact, high-quality representations. Ablation studies showed that the QRG benefits from a 3-layer architecture, while the CRG performs best with four layers, underscoring the potential of quantum-enhanced approaches in particle physics.

8 Limitations

Future work could focus on reducing parameter complexity to streamline model efficiency, try other state-of-the-art encoder networks like LorentzNet (Gong et al., 2022) and Lorentz-EQGNN (Jahin et al., 2025), and also the hybridization of RG. We also plan to extend the model to other high-energy physics tasks, such as anomaly detection in particle collisions and event reconstruction. We look forward to improving the explainability of GCL and exploring how retrospective and introspective learning in rationale discovery can guide discrimination tasks and enhance the generalization of backbone models.

References

Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Shahnawaz Ahmed, Vishnu Ajith, M. Sohaib Alam, Guillermo Alonso-Linaje, B. AkashNarayanan, Ali Asadi, Juan Miguel Arrazola, Utkarsh Azad, Sam Banning, Carsten Blank, Thomas R Bromley, Benjamin A. Cordier, Jack Ceroni, Alain Delgado, Olivia Di Matteo, Amintor Dusko, Tanya Garg, Diego Guala, Anthony Hayes, Ryan Hill, Aroosa Ijaz, Theodor Isacsson, David Ittah, Soran Jahangiri, Prateek Jain, Edward Jiang, Ankit Khandelwal, Korbinian Kottmann, Robert A. Lang, Christina Lee, Thomas Loke, Angus Lowe, Keri McKiernan, Johannes Jakob Meyer, J. A. Montañez-Barrera, Romain Moyard, Zeyue Niu, Lee James O'Riordan, Steven Oud, Ashish Panigrahi, Chae-Yeun Park, Daniel Polatajko, Nicolás Quesada, Chase Roberts, Nahum Sá, Isidor Schoch, Borun Shi, Shuli Shu, Sukin Sim, Arshpreet Singh, Ingrid Strandberg, Jay Soni, Antal Száva, Slimane Thabet, Rodrigo A. Vargas-Hernández, Trevor Vincent, Nicola Vitucci, Maurice Weber, David Wierichs, Roeland Wiersema, Moritz Willmann, Vincent Wong, Shaoming Zhang, and Nathan Killoran. Pennylane: Automatic differentiation of hybrid quantum-classical computations, 2018.

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi Jaakkola. Invariant Rationalization. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1448–1458. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chang20c.html.

Rumen Dangovski, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljacic. Equivariant self-supervised learning: Encouraging equivariance in representations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gKLAAfiytI.

Barry M. Dillon, Gregor Kasieczka, Hans Olischlager, Tilman Plehn, Peter Sorrenson, and Lorenz Vogel. Symmetries, safety, and self-supervision. *SciPost Physics*, 12(6):188, June 2022. ISSN 2542-4653. doi: 10.21468/SciPostPhys.12.6.188. URL https://www.scipost.org/10.21468/SciPostPhys.12.6.188?acad_field_slug=all.

Asitha Kottahachchi Kankanamge Don and Ibrahim Khalil. Q-SupCon: Quantum-Enhanced Supervised Contrastive Learning Architecture within the Representation Learning Framework. *ACM Transactions on Quantum Computing*, April 2024. doi: 10.1145/3660647. URL https://dl.acm.org/doi/10.1145/3660647. Just Accepted.

Roy T. Forestano, Konstantin T. Matchev, Katia Matcheva, Alexander Roman, Eyup B. Unlu, and Sarunas Verner. Accelerated discovery of machine-learned symmetries: Deriving the exceptional Lie groups $G2$, $F4$ and $E6$. *Physics Letters B*, 847:138266, December 2023a. ISSN 0370-2693. doi: 10.1016/j.physletb.2023.138266. URL https://www.sciencedirect.com/science/article/pii/S0370269323006007.

Roy T. Forestano, Konstantin T. Matchev, Katia Matcheva, Alexander Roman, Eyup B. Unlu, and Sarunas Verner. Discovering sparse representations of Lie groups with machine learning. *Physics Letters B*, 844:138086, September 2023b. ISSN 0370-2693. doi: 10.1016/j.physletb.2023.138086. URL https://www.sciencedirect.com/science/article/pii/S0370269323004203.

Roy T. Forestano, Konstantin T. Matchev, Katia Matcheva, Alexander Roman, Eyup B. Unlu, and Sarunas Verner. Identifying the group-theoretic structure of machine-learned symmetries. *Physics Letters B*, 847:138306, December 2023c. ISSN 0370-2693. doi: 10.1016/j.physletb.2023.138306. URL https://www.sciencedirect.com/science/article/pii/S0370269323006408.

Roy T. Forestano, Marçal Comajoan Cara, Gopal Ramesh Dahale, Zhongtian Dong, Sergei Gleyzer, Daniel Justice, Kyoungchul Kong, Tom Magorsch, Konstantin T. Matchev, Katia Matcheva, and Eyup B. Unlu. A Comparison between Invariant and Equivariant Classical and Quantum Graph Neural Networks. *Axioms*, 13(3):160, March 2024. ISSN 2075-1680. doi: 10.3390/axioms13030160. URL https://www.mdpi.com/2075-1680/13/3/160.

Shiqi Gong, Qi Meng, Jue Zhang, Huilin Qu, Congqiao Li, Sitian Qian, Weitao Du, Zhi-Ming Ma, and Tie-Yan Liu. An efficient Lorentz equivariant graph neural network for jet tagging. *Journal of High Energy Physics*, 2022(7):30, July 2022. ISSN 1029-8479. doi: 10.1007/JHEP07(2022)030. URL https://doi.org/10.1007/JHEP07(2022)030.

Vojtech Havlícek, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nat.*, 567(7747):209–212, 2019. doi: 10.1038/S41586-019-0980-2. URL https://doi.org/10.1038/s41586-019-0980-2.

B. Jaderberg, L. W. Anderson, W. Xie, S. Albanie, M. Kiffner, and D. Jaksch. Quantum self-supervised learning. *Quantum Science and Technology*, 7(3):035005, May 2022. ISSN 2058-9565. doi: 10.1088/2058-9565/ac6825. URL https://dx.doi.org/10.1088/2058-9565/ac6825. Publisher: IOP Publishing.

Md Abrar Jahin, Md Sakib Hossain Shovon, Md Saiful Islam, Jungpil Shin, Muhammad Firoz Mridha, and Yuichi Okuyama. Qamplifynet: pushing the boundaries of supply chain backorder prediction using interpretable hybrid quantum-classical neural network. *Scientific Reports*, 13(1):18246, 2023.

Md Abrar Jahin, Md. Akmol Masud, Md Wahiduzzaman Suva, M. F. Mridha, and Nilanjan Dey. Lorentz-Equivariant Quantum Graph Neural Network for High-Energy Physics. *IEEE Transactions on Artificial Intelligence*, pp. 1–11, 2025. doi: 10.1109/TAI.2025.3554461.

Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D. Nation, Lev S. Bishop, Andrew W. Cross, Blake R. Johnson, and Jay M. Gambetta. Quantum computing with Qiskit, 2024.

Qirui Ji, Jiangmeng Li, Jie Hu, Rui Wang, Changwen Zheng, and Fanjiang Xu. Rethinking dimensional rationale in graph contrastive learning from causal perspective. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 12810–12820. AAAI Press, 2024. doi: 10.1609/AAAI.V38I11.29177. URL https://doi.org/10.1609/aaai.v38i11.29177.

Keliang Jia, Fanxu Meng, and Jing Liang. Hierarchical graph contrastive learning framework based on quantum neural networks for sentiment analysis. *Inf. Sci.*, 690:121543, 2025. doi: 10.1016/J.INS.2024.121543. URL https://doi.org/10.1016/j.ins.2024.121543.

Roman Kogler, Benjamin Nachman, Alexander Schmidt, Lily Asquith, Emma Winkels, Mario Campanelli, Chris Delitzsch, Philip Harris, Andreas Hinzmann, Deepak Kar, Christine McLean, Justin Pilot, Yuta Takahashi, Nhan Tran, Caterina Vernieri, and Marcel Vos. Jet substructure at the Large Hadron Collider. *Reviews of Modern Physics*, 91(4):045003, December 2019. doi: 10.1103/RevModPhys.91.045003. URL https://link.aps.org/doi/10.1103/RevModPhys.91.045003. Publisher: American Physical Society.

Patrick T. Komiske, Eric M. Metodiev, and Jesse Thaler. Energy flow networks: deep sets for particle jets. *Journal of High Energy Physics*, 2019(1):121, January 2019. ISSN 1029-8479. doi: 10.1007/JHEP01(2019)121. URL https://doi.org/10.1007/JHEP01(2019)121.

Asitha Kottahachchi Kankanamge Don, Ibrahim Khalil, and Mohammed Atiquzzaman. A Fusion of Supervised Contrastive Learning and Variational Quantum Classifiers. *IEEE Transactions on Consumer Electronics*, 70(1): 770–779, February 2024. ISSN 1558-4127. doi: 10.1109/TCE.2024.3351649. Conference Name: IEEE Transactions on Consumer Electronics.

Martín Larocca, Nathan Ju, Diego García-Martín, Patrick J. Coles, and Marco Cerezo. Theory of overparametrization in quantum neural networks. *Nature Computational Science*, 3(6):542–551, June 2023. ISSN 2662-8457. doi: 10.1038/s43588-023-00467-6. URL https://www.nature.com/articles/s43588-023-00467-6. Publisher: Nature Publishing Group.

Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nat.*, 521(7553):436–444, 2015. doi: 10.1038/NATURE14539. URL https://doi.org/10.1038/nature14539.

Sihang Li, Xiang Wang, An Zhang, Yingxin Wu, Xiangnan He, and Tat-Seng Chua. Let Invariant Rationale Discovery Inspire Graph Contrastive Learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 13052–13065. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/li22v.html.

Ishan Misra and Laurens van der Maaten. Self-Supervised Learning of Pretext-Invariant Representations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6706–6716, 2020. doi: 10.1109/CVPR42600.2020.00674.

Yanhu Mo, Xiao Wang, Shaohua Fan, and Chuan Shi. Graph Contrastive Invariant Learning from the Causal Perspective. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8904–8912, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i8.28738. URL https://ojs.aaai.org/index.php/AAAI/article/view/28738. Number: 8.

Anupama Padha and Anita Sahoo. QCLR: Quantum-LSTM contrastive learning framework for continuous mental health monitoring. *Expert Systems with Applications*, 238:121921, March 2024. ISSN 0957-4174. doi: 10.1016/j.eswa.2023.121921.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Tianhao Peng, Xuhong Li, Haitao Yuan, Yuchen Li, and Haoyi Xiong. SOLA-GCL: Subgraph-Oriented Learnable Augmentation Method for Graph Contrastive Learning. In Toby Walsh, Julie Shah, and Zico Kolter (eds.), *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pp. 19875–19883. AAAI Press, 2025. doi: 10.1609/AAAI.V39I19.34189. URL https://doi.org/10.1609/aaai.v39i19.34189.

Huilin Qu and Loukas Gouskos. Jet tagging via particle clouds. *Physical Review D*, 101(5):056019, March 2020. doi: 10.1103/PhysRevD.101.056019. URL https://link.aps.org/doi/10.1103/PhysRevD.101.056019. Publisher: American Physical Society.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748.

Petar Veličković. Everything is connected: Graph neural networks. *Current Opinion in Structural Biology*, 79:102538, April 2023. ISSN 0959-440X. doi: 10.1016/j.sbi.2023.102538. URL https://www.sciencedirect.com/science/article/pii/S0959440X2300012X.

Hanrui Wang, Zhiding Liang, Jiaqi Gu, Zirui Li, Yongshan Ding, Weiwen Jiang, Yiyu Shi, David Z. Pan, Frederic T. Chong, and Song Han. Torchquantum case study for robust quantum circuits. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392174. doi: 10.1145/3508352.3561118. URL https://doi.org/10.1145/3508352.3561118.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML'20*, pp. 9929–9939. JMLR.org, July 2020.

Xiaofeng Wang, Shuaiming Lai, Shuailei Zhu, Yuntao Chen, Laishui Lv, and Yuanyuan Qi. Graph Anomaly Detection via Multiscale Contrastive Self-Supervised Learning From Local to Global. *IEEE Trans. Comput. Soc. Syst.*, 12(2): 485–497, 2025a. doi: 10.1109/TCSS.2024.3457161. URL https://doi.org/10.1109/TCSS.2024.3457161.

Zehong Wang, Donghua Yu, Shigen Shen, Shichao Zhang, Huawen Liu, Shuang Yao, and Maozu Guo. Select Your Own Counterparts: Self-Supervised Graph Contrastive Learning With Positive Sampling. *IEEE Trans. Neural Networks Learn. Syst.*, 36(4):6858–6872, 2025b. doi: 10.1109/TNNLS.2024.3388424. URL https://doi.org/10.1109/TNNLS.2024.3388424.

Ying-Xin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat seng Chua. Discovering invariant rationales for graph neural networks. In *ICLR*, 2022.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021. doi: 10.1109/TNNLS.2020.2978386.

Hanyu Xuan, Zhiliang Wu, Jian Yang, Bo Jiang, Lei Luo, Xavier Alameda-Pineda, and Yan Yan. Robust Audio-Visual Contrastive Learning for Proposal-Based Self-Supervised Sound Source Localization in Videos. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(7):4896–4907, 2024. doi: 10.1109/TPAMI.2024.3363508. URL https://doi.org/10.1109/TPAMI.2024.3363508.

Nobuyuki Yoshioka, Tsuyoshi Okubo, Yasunari Suzuki, Yuki Koizumi, and Wataru Mizukami. Hunting for quantum-classical crossover in condensed matter problems. *npj Quantum Information*, 10(1):1–10, April 2024. ISSN 2056-6387. doi: 10.1038/s41534-024-00839-4. URL https://www.nature.com/articles/s41534-024-00839-4. Publisher: Nature Publishing Group.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pp. 5812–5823, Red Hook, NY, USA, December 2020. Curran Associates Inc. ISBN 978-1-71382-954-6.

Da Zheng, Minjie Wang, Quan Gan, Zheng Zhang, and George Karypis. Learning graph neural networks with deep graph library. In *Companion Proceedings of the Web Conference 2020*, WWW '20, pp. 305–306, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450370240. doi: 10.1145/3366424.3383111. URL https://doi.org/10.1145/3366424.3383111.

A Dataset and Preprocessing

A.1 Details of Feature Engineering

Additional kinematic variables are derived from the original features $(p_{T,\alpha}^{(i)}, y_\alpha^{(i)}, \psi_\alpha^{(i)})$ using the 'Particle' package to enhance the model's ability to learn from the data. These engineered features include transverse mass, energy, and Cartesian momentum components, providing a more complete description of each particle's dynamics. Specifically, the transverse mass per multiplicity $(m_{\alpha,T}^{(i)})$ of particle i in jet α is calculated as:

$$m_{\alpha,T}^{(i)} = \sqrt{m_\alpha^{(i)^2} + p_{\alpha,T}^{(i)^2}} \tag{7}$$

where m is the rest mass of the particle and $p_T^{(i)}$ is its transverse momentum. Energy per multiplicity ($E_\alpha^{(i)}$) of particle i is computed using:

$$E_\alpha^{(i)} = m_{\alpha,T}^{(i)} \cosh y_\alpha^{(i)} \tag{8}$$

Kinematic momenta components per multiplicity ($\vec{p}_\alpha^{(i)} = (p_{x,\alpha}^{(i)}, p_{y,\alpha}^{(i)}, p_{z,\alpha}^{(i)})$) are derived from:

$$p_{x,\alpha}^{(i)} = p_{T,\alpha}^{(i)} \cos\psi_\alpha^{(i)}, p_{y,\alpha}^{(i)} = p_{T,\alpha}^{(i)} \sin\psi_\alpha^{(i)}, p_{z,\alpha}^{(i)} = m_{T,\alpha}^{(i)} \sinh y_\alpha^{(i)} \tag{9}$$

These components decompose the momentum of each particle into Cartesian coordinates, providing additional features for analysis.

The original and derived features are combined into an enriched feature set for each particle, defined as:

$$h_\alpha^{(i)} = \left\{ p_{T,\alpha}^{(i)}, y_\alpha^{(i)}, \psi_\alpha^{(i)}, m_{T,\alpha}^{(i)}, E_\alpha^{(i)}, p_{x,\alpha}^{(i)}, p_{y,\alpha}^{(i)}, p_{z,\alpha}^{(i)} \right\}, \tag{10}$$

where $h_\alpha^{(i)}$ represents the feature vector for particle i in jet α. We further calculate aggregate kinematic properties for each jet using the individual particle features. The total momentum vector of a jet ($\vec{p}_\alpha$) is obtained by summing the momentum components of its constituent particles:

$$\vec{p}_\alpha = \sum_i \vec{p}_\alpha^{(i)}, \tag{11}$$

with the transverse momentum of the jet ($p_{T,\alpha}$) calculated as:

$$p_{T,\alpha} = \sqrt{\left(\sum_i p_{x,\alpha}^{(i)}\right)^2 + \left(\sum_i p_{y,\alpha}^{(i)}\right)^2}, \tag{12}$$

which measures the momentum of the jet perpendicular to the beam axis. The jet mass (m_α) and rapidity (η_α) are defined as:

$$m_\alpha = \sqrt{(E_\alpha^2 - |\vec{p}_\alpha|^2)}, \eta_\alpha = \frac{1}{2} \ln\left(\frac{E_\alpha + p_{z,\alpha}}{E_\alpha - p_{z,\alpha}}\right), \tag{13}$$

where E_α is the sum of the energies of the jet's constituent particles and $p_{z,\alpha}$ is the component of the jet's momentum along the beam axis.

A.2 Details of Graph-Based Augmentation and Contrastive Learning

Next, the preprocessed graph data creates pairs or "views" as input for our CL framework. In CL, pairs of similar and dissimilar views are generated to help the model learn discriminative representations. Positive views are created by taking a graph and generating an augmented version of it, such as applying transformations like node dropping, edge perturbation, or feature masking. For instance, an augmented view of a quark jet remains labeled as similar (1), while a dissimilar pair may consist of a quark jet and a gluon jet, labeled as 0. The differentiation between positive and negative pairs is established solely through the loss function, with the model lacking an inherent understanding of the concept of 'view'. The loss function guides the model toward clustering similar samples in proximity. Figure 3b shows positive and negative pairs created for our CL process. The distorting jets method was applied to shift the positions of the jet constituents independently, with shifts drawn from a normal distribution. The shift is applied to each constituent's η and ψ values, scaled by their p_T, ensuring that lower p_T particles experience more significant shifts. The collinear fill technique added collinear splittings to the jets, filling zero-padded entries by splitting existing particles. A random proportion is applied for each selected particle to create two new particles that share the original momentum and position information.

A.2.1 Enforcing Infrared and Collinear Safety

To ensure that our methodology adheres to the principles of infrared and collinear (IRC) safety, we follow the guidelines established by Dillon et al. (Dillon et al., 2022) to avoid sensitivities to soft and collinear emissions, which are irrelevant to the physical properties of jets. Infrared safety is maintained by applying small perturbations to the η' and ψ' of soft particles. These perturbations follow normal distributions, $\eta' \sim \mathcal{N}(\eta, \frac{\Lambda_{\text{soft}}}{p_T})$ and $\psi' \sim \mathcal{N}(\psi, \frac{\Lambda_{\text{soft}}}{p_T})$, where $\Lambda_{\text{soft}} = 100\,\text{MeV}$ and p_T is the transverse momentum of the jet. Collinear safety is ensured by requiring that the sum of the transverse momenta of two collinear particles equals the total transverse momentum of the jet: $p_{T,\alpha} + p_{T,\beta} = p_T$. Additionally, the η and ψ of the two particles are kept identical, i.e., $\eta_a = \eta_b = \eta$ and $\phi_a = \phi_b = \phi$, preserving the collinear structure of the jet during training and testing. Techniques like node dropping and edge perturbation alter the graph's structure by randomly removing or changing connections between nodes. This helps to train the model with varying graph structures, ensuring it can adapt to different particle distributions and topologies.

B Details of Proposed QRGCL

B.1 Details of Encoder Network

We used the ParticleNet (Qu & Gouskos, 2020) model as our encoder to convert the augmented views of input particle features into low-dimensional embeddings. ParticleNet is a graph-based neural network optimized for jet tagging, leveraging dynamic graph convolutional neural networks (DGCNN) to process unordered sets of particles, treating jets as particle clouds.

The input to the encoder is a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n represents the number of particles and d the feature dimension (e.g., momentum, energy). ParticleNet constructs a k-nearest neighbor (k-NN) graph, connecting each particle to its k closest neighbors in feature space. The EdgeConv block begins by computing the k nearest neighbors using the particles' spatial coordinates. Edge features are constructed based on the feature vectors of these neighbors. The core operation of EdgeConv is a 3-layer multi-layer perceptron (MLP), where each layer consists of a linear transformation, batch normalization, and a ReLU activation. To improve information flow and avoid vanishing gradients, a shortcut connection inspired by the general ResNet architecture runs parallel to the EdgeConv operation, allowing input features to bypass the convolution layers. The two main hyperparameters of each EdgeConv block are k, the number of nearest neighbors, and $C = (C_1, C_2, C_3)$, the number of units in each MLP layer.

ParticleNet's architecture consists of three EdgeConv blocks. In the first block, distances between particles are computed in the pseudorapidity-azimuth (η, ψ) plane. In the following blocks, learned feature vectors from the previous layers serve as the coordinates. The number of nearest neighbors k is 16 across all blocks. The channel configurations C are (64, 64, 64), (128, 128, 128), and (256, 256, 256), respectively, indicating the units per MLP layer. After the EdgeConv blocks, global average pooling aggregates the learned features from all particles into a single vector. This vector is passed through a fully connected layer with 256 units and a ReLU activation. A dropout layer with a 0.1 probability is applied to prevent overfitting before the final fully connected layer. The output layer, with two units and a softmax function, produces the jet-level embeddings. These embeddings are then weighted according to node importance scores, emphasizing the most relevant particles. A global mean pooling operation is used further to aggregate the weighted features into a fixed-size jet representation.

B.2 Details of Quantum-Enhanced Contrastive Loss

QRGCL model uses a carefully designed loss function that integrates multiple elements: InfoNCE (van den Oord et al., 2018), alignment, uniformity, rationale-aware loss (RA loss), and contrastive pair loss (CP loss), to optimize the learning of quantum-enhanced embeddings. In this section, we provide detailed derivations and theoretical interpretations for these components. The overall objective is designed to learn discriminative graph embeddings by contrasting different views derived from graph rationales and their complements, while ensuring desirable geometric properties in the embedding space.

B.2.1 Core Contrastive Losses: InfoNCE, RA, and CP

These losses form the foundation of the contrastive learning process in QRGCL, aiming to distinguish between positive pairs (derived from similar rationales or views) and negative pairs (dissimilar rationales, complements, or other instances).

InfoNCE Loss The InfoNCE loss (van den Oord et al., 2018) serves as a general contrastive objective. In our context, it can be applied to augmented views of the entire graph or specific components. It maximizes a lower bound on the mutual information between two views, represented by their embeddings $\mathbf{z}$ and $\mathbf{z}'$. For a batch of N pairs, it is defined as:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}'_i)/T)}{\sum_{j=1}^{N} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}'_j)/T)} \right) \tag{14}$$

where $(\mathbf{z}_i, \mathbf{z}'_i)$ is a positive pair of embeddings (e.g., from two augmentations of the same graph), $\mathbf{z}'_j$ are embeddings from other instances (negatives) in the batch, $T > 0$ is the temperature hyperparameter, and $\text{sim}(\cdot, \cdot)$ denotes the cosine similarity function $\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}^\top \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$. Minimizing $\mathcal{L}_{\text{InfoNCE}}$ encourages the embeddings of positive pairs to be more similar than the embeddings of negative pairs.

Derivation of InfoNCE Loss Let $\mathcal{Z} = (\mathbf{z}_i, \mathbf{z}'_i)_{i=1}^{N}$ be a batch of N positive contrastive pairs. Assume that the joint distribution $p(\mathbf{z}, \mathbf{z}')$ is known, and our goal is to maximize the mutual information $\mathcal{I}(\mathbf{z}, \mathbf{z}')$. Using the

Donsker-Varadhan representation of KL divergence:

$$\mathcal{I}(\mathbf{z}, \mathbf{z}') \geq \mathbb{E}_{p(\mathbf{z},\mathbf{z}')} \left[\log \frac{f(\mathbf{z}, \mathbf{z}')}{\mathbb{E}_{p(\mathbf{z})p(\mathbf{z}')}[f(\mathbf{z}, \mathbf{z}')]} \right] \tag{15}$$

Choose $f(\mathbf{z}, \mathbf{z}') = \exp(\text{sim}(\mathbf{z}, \mathbf{z}')/T)$. Replacing the denominator with a sum over negatives in the batch (empirical estimate), we get the InfoNCE bound:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{e^{\text{sim}(\mathbf{z}_i, \mathbf{z}_i')/T}}{\sum_{j=1}^{N} e^{\text{sim}(\mathbf{z}_i, \mathbf{z}_j')/T}} \right) \tag{16}$$

This lower bounds the mutual information $\mathcal{I}(\mathbf{z}, \mathbf{z}')$, making it suitable for contrastive representation learning.

RA Loss The RA loss specifically focuses on contrasting positive pairs derived from the graph's "rationale" (critical subgraph identified for classification) against other rationale-derived pairs within the batch. Let $\mathbf{z}_1^i$ and $\mathbf{z}_2^i$ be embeddings corresponding to two different views or augmentations of the rationale of the i-th graph in a batch of size N. The RA loss aims to make $\mathbf{z}_1^i$ similar to $\mathbf{z}_2^i$ while distinguishing it from $\mathbf{z}_2^j$ for $j \neq i$. The loss is calculated as:

$$\mathcal{L}_{\text{RA}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}}{\sum_{j=1}^{N} e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^j)/T)} - e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}} \right) \tag{17}$$

Here, the denominator sums the similarity scores between the anchor rationale embedding $\mathbf{z}_1^i$ and all rationale embeddings $\mathbf{z}_2^j$ from the second view in the batch, excluding the positive pair similarity itself. This forces the model to learn representations that are highly specific to the corresponding rationale pairs.

Derivation of RA Loss We reinterpret the RA loss as a softmax-based ranking loss. Let $P(i|j)$ denote the probability of matching rationale pair (i, j):

$$P(i|j) = \frac{e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}}{\sum_{k=1}^{N} e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^j)/T)}} \tag{18}$$

To ensure the model doesn't trivially match a pair to itself, we subtract the matching term from the denominator:

$$\mathcal{L}_{\text{RA}} = -\log P(i|j) = -\log \left(\frac{e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}}{\sum_{j=1}^{N} e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^j)/T)} - e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}} \right) \tag{19}$$

This form can be derived from maximizing a modified log-likelihood over rationale-based matching while excluding the anchor-positive redundancy from the normalization.

CP Loss The CP loss introduces the concept of a "complement" view (pairs involving nodes not deemed crucial for the classification task), derived from the non-rationale parts of the graph. It encourages the rationale embedding $\mathbf{z}_1^i$ to be similar to its corresponding rationale pair $\mathbf{z}_2^i$, while simultaneously being dissimilar to embeddings derived from the complement regions, denoted by $\mathbf{z}_3^j$. This helps the model distinguish between critical (rationale) and non-critical (complement) information:

$$\mathcal{L}_{\text{CP}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}}{\sum_{j=1}^{N} e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_3^j)/T)} + e^{(\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)/T)}} \right) \tag{20}$$

where the denominator includes the sum of similarities between the anchor rationale $\mathbf{z}_1^i$ and all complement embeddings $\mathbf{z}_3^j$ from the third view, plus the positive pair similarity. This loss helps the model differentiate genuine relationships between similar samples from spurious correlations by learning from pairs with varying significance.

Theorem 1 (Interpretation of RA and CP Losses)**.** *Minimizing the combined loss $\mathcal{L}_{RA} + \lambda \mathcal{L}_{CP}$ encourages the model to learn representations $\mathbf{z}$ such that:*

1. *Embeddings $\mathbf{z}_1^i$ and $\mathbf{z}_2^i$ derived from the same rationale are pulled closer together in the embedding space.*
2. *Embeddings $\mathbf{z}_1^i$ derived from one rationale are pushed apart from embeddings $\mathbf{z}_2^j$ (for $j \neq i$) derived from other rationales.*

3. *Embeddings $\mathbf{z}_1^i$ derived from a rationale are pushed apart from embeddings $\mathbf{z}_3^j$ derived from complement regions.*

This promotes learning of features that are both specific to the rationale's identity and distinct from non-critical graph components.

Proof Sketch. The structure of $\mathcal{L}_{\text{RA}}$ equation 17 and $\mathcal{L}_{\text{CP}}$ equation 20 follows the standard contrastive loss form $-\log(\frac{\text{positive}}{\text{positive}+\sum\text{negatives}})$. Minimizing this loss is equivalent to maximizing the log-probability of correctly identifying the positive pair among a set of negatives. For $\mathcal{L}_{\text{RA}}$, the "negatives" are other rationale pairs within the batch ($\mathbf{z}_2^j, j \neq i$). Minimization increases $\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)$ relative to $\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^j)$, thus pulling positive rationale pairs together and pushing them apart from other rationale pairs. For $\mathcal{L}_{\text{CP}}$, the "negatives" are the complement embeddings ($\mathbf{z}_3^j$). Minimization increases $\text{sim}(\mathbf{z}_1^i, \mathbf{z}_2^i)$ relative to $\text{sim}(\mathbf{z}_1^i, \mathbf{z}_3^j)$, thus pushing rationale embeddings away from complement embeddings. Combining these objectives achieves the stated properties. □

B.2.2 Geometric Regularization Losses: Alignment and Uniformity

These losses, inspired by (Wang & Isola, 2020), aim to improve the quality of the embedding space by enforcing desirable geometric properties, preventing representational collapse, and improving feature diversity.

Alignment Loss The alignment loss measures the expected distance between normalized embeddings of positive pairs (p_{pos}), encouraging them to map to nearby points in the embedding space. Assuming the input embeddings $\mathbf{z}_1$ and $\mathbf{z}_2$ are L_2-normalized (denoted $\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2$), the alignment loss is defined as the expected squared Euclidean distance (L^2):

$$\mathcal{L}_{\text{align}} \triangleq \mathbb{E}_{(\hat{\mathbf{z}}_1,\hat{\mathbf{z}}_2)\sim p_{\text{pos}}} \left[\|\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2\|_2^2\right] \tag{21}$$

where p_{pos} is the distribution of positive pairs and normalization ensures embeddings lie on the unit hypersphere. Minimizing this loss forces the representations of augmented views of the same input to be identical, promoting invariance.

Quantum Fidelity Alignment (Theoretical) As a theoretical alternative motivated by quantum information, we used *quantum state fidelity* as a distance metric, replacing the traditional L^2 distance typically used in classical CL. Quantum fidelity measures the closeness between two quantum states. If embeddings $\mathbf{z}_1, \mathbf{z}_2$ represent quantum states via density matrices ρ_1, ρ_2, the fidelity-based alignment loss is:

$$\mathcal{L}_{\text{align}} \triangleq \mathbb{E}_{(\rho_1,\rho_2)\sim p_{\text{pos}}}[1 - \mathcal{F}(\rho_1, \rho_2)] = \mathbb{E}_{(\rho_1,\rho_2)\sim p_{\text{pos}}} \left[1 - \left(\text{Tr}\left(\sqrt{\sqrt{\rho_1}\rho_2\sqrt{\rho_1}}\right)\right)^2\right] \tag{22}$$

where $\mathcal{F}(\rho_1, \rho_2)$ is the fidelity between states ρ_1 and ρ_2. Lower values of $1-\mathcal{F}$ indicate higher similarity between quantum states. If $\rho_i = |\psi_i\rangle\langle\psi_i|$ are pure states derived from normalized vectors $\mathbf{z}_i$, then $\mathcal{F}(\rho_1, \rho_2) = |\langle\psi_1|\psi_2\rangle|^2 = \cos^2(\theta)$. Therefore, alignment loss becomes:

$$\mathcal{L}_{\text{align}} \triangleq \mathbb{E}_{(\rho_1,\rho_2)\sim p_{\text{pos}}} [1 - \mathcal{F}(\rho_1, \rho_2)] = \mathbb{E}_{(\rho_1,\rho_2)} \left[1 - |\langle\psi_1|\psi_2\rangle|^2\right] \tag{23}$$

This aligns the quantum embeddings up to a global phase, a desirable property in quantum feature spaces.

Uniformity Loss The uniformity loss encourages the embeddings to be uniformly distributed on the unit hypersphere. This prevents the model from collapsing all embeddings to a single point or small region, thereby preserving the discriminative information contained in the representations. It is defined as the expected log pairwise potential over all distinct data points:

$$\mathcal{L}_{\text{uniform}} \triangleq \log \mathbb{E}_{(\mathbf{z}_x,\mathbf{z}_y)\sim p_{\text{data}}, x\neq y} \left[e^{-t\|\mathbf{z}_x - \mathbf{z}_y\|_2^2}\right] \tag{24}$$

where p_{data} is the distribution of data samples, and $t > 0$ is a hyperparameter (typically $t = 2$). Minimizing this loss encourages larger distances between embeddings of different samples, promoting uniformity.

Theorem 2 (Role of Alignment and Uniformity)**.** *Minimizing the combined loss $\alpha\mathcal{L}_{align} + \beta\mathcal{L}_{uniform}$ regularizes the embedding space by:*

1. *Enforcing invariance to data augmentations or view generation (Alignment).*
2. *Maximizing the entropy of the embedding distribution on the unit hypersphere, preserving feature diversity (Uniformity).*

These properties contribute to learning higher quality, more discriminative representations.

Proof Sketch. Minimizing $\mathcal{L}_{\text{align}}$ equation 21 directly minimizes the distance between positive pairs, achieving local invariance. Minimizing $\mathcal{L}_{\text{uniform}}$ equation 24 minimizes the potential energy of a system where points repel each other via a Gaussian kernel e^{-td^2}, leading to a uniform distribution on the embedding manifold (unit hypersphere if normalized) (Wang & Isola, 2020). Uniformity is related to maximizing the entropy of the representations, thus preserving information from the input data. □

B.2.3 Overall QRGCL Objective

The overall loss for the QRGCL model is a weighted combination of the InfoNCE, RA, and CP loss, with optional contributions from the alignment and uniformity terms, allowing for flexible control over the learning process:

$$\mathcal{L}_{\text{QRGCL}} = \mathcal{L}_{\text{RA}} + \lambda\mathcal{L}_{\text{CP}} + \alpha\mathcal{L}_{\text{align}} + \beta\mathcal{L}_{\text{uniform}} + \delta\mathcal{L}_{\text{InfoNCE}} \tag{25}$$

where $\lambda, \alpha, \beta, \delta \geq 0$ are hyperparameters balancing the contribution of each loss component. During experimentation, the uniformity term ($\beta\mathcal{L}_{\text{uniform}}$) might be omitted ($\beta = 0$) if it hinders performance empirically.

Remark 1. *Each component has a bounded gradient and differentiable form, ensuring compatibility with stochastic gradient descent. Further, RA and CP are mutually reinforcing, and the inclusion of alignment and uniformity ensures geometric and quantum-consistent embeddings.*

C Details of Benchmark Models

We developed two classical models (GNN and EQGNN), three quantum models (QGNN, EQGNN, and QCL), five hybrid classical-quantum models (CQCL, 3 QCGCL variants, and QRGCL with $RX + H$ encoding) (see Table 3), and the classical counterpart of QRGCL, i.e., CRGCL (see Table 2).

C.1 Classical RGCL (CRGCL)

The classical RG (CRG) of RGCL is a GNN estimator designed to generate node representations from graph-structured data, which acts as the counterpart of QRGCL. It consists of three graph convolutional layers (GCN) that reduce feature dimensionality from the input to 32, 16, and finally, 8. Each layer is followed by batch normalization to stabilize training and reduce internal covariate shifts. ReLU activation is applied after the first two layers, while dropout is used to mitigate overfitting. The final output passes through a linear layer to yield a single value per node, followed by a softmax for probabilistic representation. For a fair comparison between the CRG and QRG, we use the well-established ParticleNet as the encoder network, followed by the same projection head used in QRGCL.

C.2 GNN

The architecture consists of layers where the message-passing mechanism updates node features based on neighboring nodes and their relationships. An MLP processes the node features in each layer, while an edge MLP aggregates the edge attributes. After several layers, the updated node features are pooled to form a graph-level representation, passing through a fully connected neural network for classification. GNNs can inherently handle the permutation of nodes due to their graph-based nature.

C.3 Equivariant GNN (EGNN)

EGNNs extend GNNs by incorporating symmetry transformations, such as rotational and translational equivariance. This means that the model's predictions remain consistent under transformations of the input data. The coordinates of the nodes are updated during each layer based on interactions with neighboring nodes. The model utilizes additional features like distance metrics between node pairs to enforce equivariance in graph processing.

C.4 Quantum GNN (QGNN)

In QGNN, the graph's node features are embedded into quantum states using an embedding layer. These quantum states evolve under a parameterized Hamiltonian, which encodes the interactions between nodes (qubits) based on the graph's adjacency matrix. The QGNN model relies on unitary transformations to evolve the quantum state over multiple layers. After the final layer, quantum measurements are performed, and the results are passed through a fully connected layer to make predictions.

C.5 Equivariant Quantum GNN (EQGNN)

EQGNNs (Jahin et al., 2025; Forestano et al., 2024; 2023a;b;c) are quantum analogs of EGNNs, incorporating equivariance into the quantum architecture. Like QGNNs, EQGNNs operate on quantum states but ensure that the learned representations are equivariant under symmetries, such as permutations or rotations. The final quantum states are aggregated through pooling, ensuring the network remains permutation-equivariant. This aggregation is followed by classical post-processing to yield the model's predictions.

C.6 QCL

QCL employs a quantum convolutional neural network (QCNN) with input size 18×18 with zero padding as the encoder, where data-reuploading circuits (DRCs) replace classical convolutional kernels. The QCNN consists of 2 similar layers: the first with a (3×3) kernel, stride 2, and a 3-qubit DRC as a filter. The final encoding size after flattening is 16. The projection network either uses a 2-node linear layer.

C.7 Classical-Quantum CL (CQCL)

Hybrid CQCL is similar to QCL except for the final projection head, which uses a $\log_2^{d=2}$ qubit (d = embedding dimension) DRC circuit. To evaluate the ability of QCL and CQCL to generate generalized representations, we make predictions using a simple MLP, an input layer of size 256, and a hidden layer of size 32, both with BatchNorm and LeakyReLU.

C.8 Quantum-Classical GCL (QCGCL)

QCGCL architecture initially uses multiple layers of graph attention-based convolution (GATConv) layers to capture graph features. Each GAT layer is followed by batch normalization and residual connections, which allow the model to skip layers for improved training. The graph-level embedding is achieved through mean and max pooling. A quantum circuit then processes this pooled GNN output. We benchmarked QCGCL using 3 different quantum encoders, as depicted in Table 3. Finally, the GNN and quantum outputs are concatenated and passed through a fully connected readout layer to generate the final prediction.