MACHINE LEARNING
Science and Technology

PAPER

Quantum complete graph self-attention network for particle flow classification

Ting Li*, **Shanglong Liu**, **GuangZhi Xu** and **Peizhong Xie**

School of Communications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, People's Republic of China

* Author to whom any correspondence should be addressed.

E-mail: lit@njupt.edu.cn

Keywords: quantum complete graph neural network, particle flow classification, quantum self-attention mechanism, quantum distributed dimensionality reduction architecture

OPEN ACCESS

RECEIVED
14 July 2025

REVISED
24 December 2025

ACCEPTED FOR PUBLICATION
2 January 2026

PUBLISHED
12 January 2026

Abstract

Particle flow classification is essential in high-energy and nuclear physics. This work proposes the quantum complete graph self-attention network (QCGAT), which selects particles with high transverse momentum and uses a quantum self-attention mechanism for feature extraction. Unlike previous methods, the proposed QCGAT framework introduces a non-measurement quantum self-attention mechanism within an entangled quantum network, enabling the direct embedding of attention coefficients into quantum state amplitudes and thereby avoiding quantum-to-classical conversion. Furthermore, this work introduces a quantum distributed dimensionality reduction architecture that specifically enables tunable cross-scale data adaptation. Experiments on the Top Quark Tagging dataset show that QCGAT achieves 83.5% accuracy and 88.7% AUC, surpassing hybrid models by 1% and 0.4%. Compared with traditional quantum encoder baseline, it improves accuracy by 0.7%; when compared with traditional quantum encoder baselines having similar or fewer parameter counts, it achieves up to 1.6% higher accuracy. These results highlight the effectiveness and potential of QCGAT in quantum machine learning for particle physics.

1. Introduction

The concept of particle flow classification originated in high-energy physics, particularly in the context of experiments conducted at large-scale particle detectors such as the Large Hadron Collider. By the late 20th century, rapid advancements in detector technologies brought about the need for more accurate reconstruction of particle types and their trajectories from massive volumes of collision data. Traditional classification techniques, which relied primarily on information from individual detector components, proved insufficient for resolving the increasingly complex and overlapping particle interactions observed in high-energy events. These limitations highlighted the need for more integrated and sophisticated approaches to particle classification.

To address the challenges associated with particle identification and reconstruction, the particle flow algorithm [1] was developed. Its core concept is to integrate information from multiple detector components to accurately reconstruct the identity and energy of individual particles. In the 21st century, particle flow classification has been propelled by two major technological advancements. First, improvements in detector technologies—such as high-resolution sensors and large-scale data acquisition systems—have enabled increasingly precise measurements of particle flows. Second, the rise of machine learning and deep learning techniques has introduced new modeling capabilities and significantly advanced the field.

Among early deep learning-based models, the particle flow network (PFN) [2], built upon the DeepSets framework, treats each particle in a jet as an independent input, thus avoiding the information loss inherent in traditional pixel-based approaches. By leveraging permutation invariance, PFN captures global dependencies among particles while remaining insensitive to input ordering. Building upon

this, ParticleNet [3] models particles as points in space, combining their spatial and physical features, and employs graph neural networks (GNNs) [4–6] to represent and learn the topological structure of particle jets. Through EdgeConv operations, the network dynamically constructs local neighborhoods, effectively learning local geometric patterns and improving the modeling of jet substructures.The introduction of GNNs has markedly advanced modeling capabilities in this domain [7–9]. More recently, attention-based models have further enhanced performance. The point cloud transformer [10] adapts the Transformer architecture to particle flow data represented as point clouds, enabling the capture of long-range dependencies and overcoming the locality limitations of GNNs. The Particle Transformer (ParT) [11] extends this concept by incorporating pairwise particle interactions directly into the attention mechanism, achieving superior tagging performance and significantly outperforming the previously state-of-the-art ParticleNet.

These models have demonstrated strong performance in key tasks such as jet event classification, background suppression, and rare particle identification, highlighting the growing role of deep learning in advancing particle flow analysis.

Quantum machine learning (QML) is an interdisciplinary field that integrates quantum computing with classical machine learning, and it has advanced rapidly in recent years due to significant progress in quantum hardware technologies. Drawing inspiration from the translation-invariant characteristics of convolutional and pooling layers in classical convolutional neural networks (CNNs), Cong *et al* proposed the quantum CNN (QCNN) [12]. In this architecture, convolutional layers apply quasi-local unitary operations, while pooling layers perform qubit measurements followed by conditional rotations. These operations are repeated in a hierarchical fashion until the quantum system is sufficiently reduced, after which a fully connected layer and final qubit measurement complete the classification process.

To address the limitations of earlier QCNN models in handling multi-channel data, Smaldone *et al* developed a QCNN architecture tailored for multi-channel supervised learning tasks [13]. In the domain of generative modeling, Hu *et al* introduced the quantum generative adversarial network (QGAN) [14] which was experimentally validated on superconducting quantum circuits. In QGAN, the quantum generator is trained adversarially to produce quantum states that closely match real data statistics, rendering them indistinguishable by the discriminator.

Beyond convolutional architectures, QML has also extended into graph-structured data. Verdon *et al* proposed the quantum GNN (QGNN) [15], enabling quantum and classical probabilistic inference over graph-structured inputs. Building on this, Hu *et al* introduced the quantum graph convolutional network [16], which employs givens rotations to perform message passing among neighboring nodes, effectively encoding adjacency information directly into quantum circuits.

Attention mechanisms have also been explored in quantum neural networks. Li *et al* proposed the quantum self-attention neural network [17], incorporating a Gaussian-projected quantum self-attention module as a quantum analog to classical attention. Chen *et al* further developed the quantum mixed-state attention network [18], which computes attention weights directly at the quantum level without quantum-to-classical conversion, thereby improving computational efficiency and accuracy.

The introduction of QML has further advanced the development of particle flow classification techniques. By harnessing the superposition and parallel processing capabilities of quantum states, QML holds the potential to overcome the computational limitations of classical approaches, particularly in high-dimensional data analysis and complex optimization tasks. Recent studies have demonstrated the preliminary application of quantum algorithms [19, 20] in high-energy physics. Notably, Casals *et al* [21] introduced a hybrid classical–quantum framework for jet tagging that employs guided compression to jointly optimize reconstruction and classification losses, thereby bridging the gap between theoretical advantages and hardware constraints. To contextualize the contributions of QML, it is essential to distinguish between theoretical guarantees and practical performance. Theoretical works have provided rigorous proofs of quantum advantages, often based on mechanisms like discrete logarithms or Grover's search [22, 23], or learning-guided separations [24]. However, these advantages are typically established on artificially constructed problems. Conversely, recent numerical studies focus on real-world HEP data, providing empirical evidence of quantum utility. For instance, Belis *et al* [25] demonstrated that quantum resources can enhance anomaly detection in latent spaces, while Zhang *et al* [26] showed the efficiency of quantum kernel methods for detecting anomalous quartic gauge couplings. These works complement theoretical foundations by validating potential advantages in practical scenarios.

Existing related studies have theoretically demonstrated that QML can enhance expressivity through data re-uploading strategies enabling universal classification with minimal quantum resources [27] and improve generalization ability even with limited training data, requiring only polynomial-sized datasets for efficient generalization in high-dimensional quantum systems [28].

The unique properties of quantum systems—especially their ability to process information in superposed and entangled states—offer the possibility of enhanced efficiency when modeling intricate particle interactions in high-dimensional feature spaces. Although current quantum hardware remains in its early stages and faces challenges related to scalability and error correction, the integration of quantum and classical machine learning has opened promising new directions for particle flow classification research [29]. This interdisciplinary approach not only improves the precision of particle identification in high-energy physics but also provides meaningful real-world applications for quantum computing technologies.

In this work, we propose quantum complete graph self-attention network (QCGAT), a quantum-native self-attention network that achieves functional equivalence to classical weighted summation but follows a distinct quantum computational pathway. Specifically, we construct an entangled quantum network to compute self-attention coefficients between particles. These coefficients are embedded directly into quantum state amplitudes and used to perform weighted summations of the corresponding particle features, enabling feature updates without quantum-to-classical conversion. To the best of our knowledge, this represents the first complete implementation of self-attention within a QML framework. Our model operates on a complete graph [30], leveraging features from a small number of particles and encoding only node information, without relying on edge features.

This process constructs a complete graph, where each particle feature corresponds to a node. A quantum self-attention mechanism is then employed to efficiently aggregate messages from neighboring nodes and dynamically update node features. This approach integrates particle-physics intuition—that key particle features encapsulate the global characteristics of the jet—with the coherent interactions enabled by quantum states. It preserves the structural modeling capabilities of classical self-attention mechanisms while leveraging quantum properties such as superposition and entanglement, thereby offering a novel paradigm for high-dimensional quantum information processing.

The remainder of this paper is organized as follows. Section 2 provides background on GNNs, self-attention mechanisms, and quantum computing. Section 3 details the design principles of the proposed QCGAT module. Section 4 introduces the overall system architecture and the implementation of quantum gate operations. Section 5 presents experimental results along with performance analysis. Finally, section 6 concludes the paper and outlines directions for future research.

2. Background

2.1. GNN

The introduction of GNNs aims to exploit the structural properties of graph-based data, such as those found in social networks, chemical compounds, or physical systems, by modeling relationships between nodes and edges. A graph is typically represented as $G(N,E)$, where N denotes a set of nodes and E the set of edges connecting them—as illustrated by the basic undirected structure shown in figure 1. If every pair of nodes in the graph is connected by an edge, the graph is referred to as a complete graph. Each node is associated with a feature vector, with the feature of the ith node denoted as $\mathbf{x}_i \in \mathbb{R}^d$, where d is the dimensionality of the feature vector.

The neighborhood of node i, denoted as $\mathcal{N}_i = \{j : e_{ij} \in E\}$, represents the set of nodes connected to i by edges. GNNs employ a message-passing paradigm to acquire and aggregate information from neighboring nodes. Specifically, the feature of each neighbor node j is first transformed via a learned function $F(\mathbf{x}_j)$, as shown in equation (1),

$$F\left(\mathbf{x}_j\right) = \mathbf{W}_j\mathbf{x}_j + \mathbf{b}. \tag{1}$$

These transformed features are then aggregated using an aggregation function $G\left(\left\{F(\mathbf{x}_j) : j \in \mathcal{N}_i\right\}\right)$, where common choices for G include summation, averaging, or max-pooling. Subsequently, the aggregated message is combined with the transformed feature of node i, denoted $F(\mathbf{x}_i)$, to obtain an intermediate representation $\mathbf{z}_i$, as described in equation (2). The combination function C may involve summation, concatenation, or other operations. Finally, a nonlinear activation function $\sigma(\mathbf{z}_i)$, such as ReLU or Softmax, is applied to compute the updated feature of node i, denoted $\mathbf{h}_i$, as shown in equation (3). This process is executed in parallel across all nodes in the graph, allowing GNNs to iteratively update node features by aggregating and transforming information from their local neighborhoods,

$$\mathbf{z}_i = C\left(F(\mathbf{x}_i),\ G\left(\left\{F\left(\mathbf{x}_j\right) : j \in \mathcal{N}_i\right\}\right)\right), \tag{2}$$

$$\mathbf{h}_i = \sigma\left(\mathbf{z}_i\right). \tag{3}$$

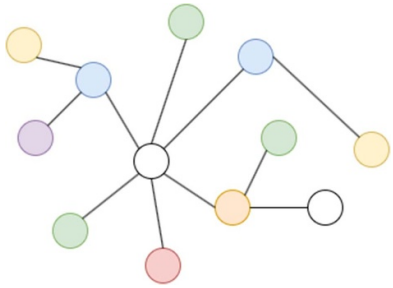

Figure 1. Graph structure:the central node is fully connected to all surrounding neighbor nodes, enabling message passing and feature aggregation within the GNN framework.

Based on the established understanding of the update mechanisms in GNNs, we consider a graph $G(N,E)$, where all N nodes are represented by a feature matrix $X \in \mathbb{R}^{N\times d}$, where d denotes the feature dimension of each node. The inter-node relationships are encoded by an adjacency matrix $A \in \mathbb{R}^{N\times N}$, where each element a_{ij} is defined as:

$$a_{ij} = \begin{cases} 1, & e_{ij} \in E \\ 0, & e_{ij} \notin E. \end{cases} \tag{4}$$

GNNs update node representations through iterative message passing mechanisms across the entire graph. In this process, each node feature is initially transformed using a learnable weight matrix $\mathbf{W} \in \mathbb{R}^{d\times k}$, where d denotes the dimensionality of the input features and k represents the dimensionality of the transformed features. This operation yields a new feature matrix $\mathbf{Z} \in \mathbb{R}^{N\times k}$ for all N nodes in the graph. The resulting features $\mathbf{Z}$ are used as the input to subsequent aggregation and non-linear transformation steps in the GNN pipeline,

$$Z = XW \in R^{N\times k}. \tag{5}$$

Subsequently, message aggregation is performed by utilizing the adjacency matrix to select the neighbors of each node and aggregate their features, resulting in the final node representation matrix Y,

$$Y = AZ = AXW \in R^{N\times k}. \tag{6}$$

In order to account for the features of each node during aggregation, a self-loop is typically added to every node in the graph structure,

$$\tilde{A} = A + I, \tag{7}$$

$$Y = \tilde{A}Z = \tilde{A}XW \in \mathbb{R}^{N\times k}. \tag{8}$$

GNNs are capable of handling node-level, edge-level, and graph-level tasks. This work focuses on a graph-level classification task based on a fully connected graph.

2.2. Self-attention mechanism

Since its introduction by Vaswani *et al* [31], the self-attention mechanism has attracted significant attention in the machine learning community due to its ability to model long-range dependencies without relying on recurrence or convolution. This innovation marked a departure from the previously dominant architectures, such as recurrent neural networks [32] and CNNs [33], which relied heavily on sequential or local processing. Self-attention—also known as intra-attention—enables each position in a sequence to attend to all other positions, allowing the model to capture global contextual information more efficiently. This mechanism has since established itself as a fundamental component in a wide range of deep learning architectures, particularly in natural language processing and vision tasks.

The self-attention mechanism can be mathematically and formally described as a process that maps a query and a set of key-value pairs to an output, where all entities—the query, keys, values, and output—are represented as vectors. The output is computed as a weighted sum of the value vectors, with each weight reflecting the relevance between the corresponding key and the query, as determined by a compatibility function. For each input, which is encoded into a feature vector $\mathbf{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n}\}$, where $0 \leqslant i \leqslant s$, $s \geqslant 1$, and $i, s \in \mathbb{Z}$ denote the input index and the total sequence length respectively, linear

transformations are applied to obtain the query vector $\mathbf{q}_i$, key vector $\mathbf{k}_i$, and value vector $\mathbf{v}_i$, as defined in equations (9)–(11). The self-attention scores are then computed by performing dot products between the query vector of each input and the key vectors of all inputs, as shown in equation (12), where $0 \leqslant i,j \leqslant s$, and the projection matrices $W_Q, W_K, W_V \in \mathbb{R}^{n\times d_k}$,

$$\mathbf{q}_i = \mathbf{x}_i W_Q, \tag{9}$$

$$\mathbf{k}_i = \mathbf{x}_i W_K, \tag{10}$$

$$\mathbf{v}_i = \mathbf{x}_i W_V, \tag{11}$$

$$\text{score}_{i,j} = \mathbf{q}_i \mathbf{k}_j^{\mathrm{T}}. \tag{12}$$

The self-attention mechanism allows the computation of attention functions across a set of queries. In this framework, the input feature vectors are organized into a matrix X, from which the query, key, and value matrices—denoted as Q, K, and V, respectively—are obtained through learned linear projections. This matrix formulation enables the self-attention mechanism to efficiently compute pairwise interactions between all positions in the input sequence simultaneously and significantly enhances computational efficiency while facilitating the modeling of global dependencies,

$$Q = XW_Q, \tag{13}$$

$$K = XW_K, \tag{14}$$

$$V = XW_V. \tag{15}$$

To stabilize gradient updates during training, the self-attention scores are scaled by the inverse square root of the dimensionality of the key vectors. This operation mitigates the risk of vanishing gradients caused by large dot-product values. Following this, the scaled scores are passed through the Softmax activation function to obtain normalized attention weights, which capture the relevance between the query and each key. These attention weights are then multiplied by the corresponding value vectors to emphasize the most informative components of the input. Finally, the weighted values are aggregated through summation to produce the output representation for each input element,

$$\text{Attention}\,(Q,K,V) = \text{Softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V. \tag{16}$$

2.3. Quantum self-attention scores

In quantum computing, the inner product between two quantum states inherently quantifies their overlap. Consider two quantum systems, each comprising n qubits, initialized in the all-zero state. These systems evolve independently through two unitary operations, denoted as U_1 and U_2, resulting in the quantum states $|\psi\rangle$ and $|\phi\rangle$, respectively:

$$\begin{aligned} |\psi\rangle &= U_1|0\rangle^{\otimes n} = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle \\ |\varphi\rangle &= U_2|0\rangle^{\otimes n} = \sum_{j=0}^{2^n-1} \beta_j |j\rangle. \end{aligned} \tag{17}$$

Let $N = 2^n$. For notational convenience, the tensor product of n qubits in the $|0\rangle$ state is abbreviated as $|0\rangle^{\otimes n} = |0\rangle_n$. The inner product between these quantum states is then given by:

$$\langle\psi|\varphi\rangle = \langle 0|_n {U_2}^{\dagger} U_1 |0\rangle_n. \tag{18}$$

Inspired by the approach in [34], we construct the quantum circuit shown in figure 2 to generate the corresponding quantum state,

$${U_2}^{\dagger} U_1 |0\rangle_n = \gamma_0 |0\rangle_n + \sum_{i=1}^{2^n-1} \gamma_i |i\rangle_n. \tag{19}$$

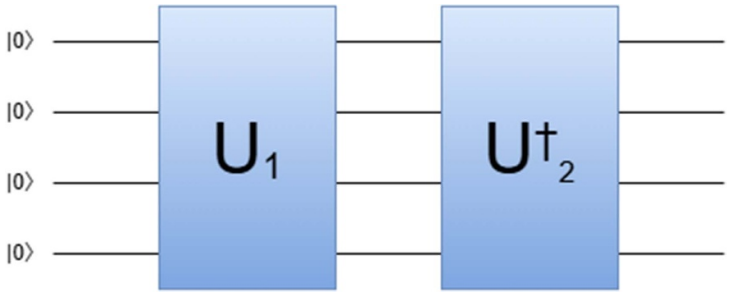

Figure 2. Quantum circuit for calculating the inner product of two quantum states. U_1: the unitary gate acts on the initial quantum state $|0\rangle^{\otimes n}$,generating the quantum state $|\psi\rangle$ through a specific unitary transformation. $U_2^{\dagger}$: the unitary gate represents the Hermitian conjugate of U_2 . It is used in the quantum circuit to compute the inner product between the two quantum states $|\psi\rangle$ and $|\varphi\rangle$.

Therefore, the value of the inner product is solely encoded in the amplitude associated with the $|0\rangle^n$ state component,

$$\langle\psi|\varphi\rangle = \gamma_0. \tag{20}$$

The inner product between quantum states serves as a self-attention coefficient, reflecting the similarity or correlation between two quantum-encoded representations. In the proposed quantum complete graph self-attention mechanism, self-attention scores between particles are computed using quantum circuits. Specifically, if a particle's query and key are encoded as quantum states $|\psi\rangle$ and $|\varphi\rangle$, respectively, their inner product serves as the self-attention score, measuring the alignment between them. As described in equations (19) and (20), particle features are first encoded into quantum states and then transformed via unitary operations implemented by quantum gates. The resulting self-attention coefficients are extracted by measuring the projection onto the $|0\rangle_n\langle 0|_n$ basis state, which corresponds to the amplitude overlap between the transformed quantum states.

3. Design principles of the QCGAT module

In particle jet classification, particle features are modeled as graph nodes, with interactions or correlations represented as edges. In this work, we focus on the four particles with the highest transverse momentum in each jet, treating them as representative of the overall jet characteristics. To capture potential correlations among these high-momentum particles, we model their relationships as a complete graph, in which all nodes are connected.

The QCGAT model proposed in this work comprises three primary components. The first is the index register, which employs Hadamard gates to generate a quantum superposition state, enabling the execution of controlled operations conditioned on distinct particle features. This register serves to differentiate individual particles and forms the foundation for propagating self-attention coefficients throughout the circuit. The second component is the self-attention register, which is designed to compute self-attention coefficients among particles. These coefficients are formulated in accordance with equation (19). By exploiting the superposition principle of the quantum states in the index register, the model is capable of simultaneously calculating all self-attention coefficients for a given particle under various control conditions imposed by the control qubits. A projection measurement onto the all-zero state is employed to isolate and retain the relevant self-attention contributions. The third component is the value register, which performs weighted summation operations. The computed self-attention coefficients are encoded in the amplitude distribution of the index register. By replicating the control configurations of the index register, the model performs weighted summation across the value components of different particles, thereby updating the particle features accordingly.

3.1. Quantum state representation of the index register

For a system comprising $N = 2^n$ particles, the index register requires n qubits to encode particle feature indices. The register is initialized in the quantum state $|0\rangle^{\otimes n}$. A uniform superposition over all particle indices is generated by applying Hadamard gates to all qubits, resulting in the quantum state:

$$|\psi_{\mathrm{ir}}\rangle = H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle^{\otimes n}. \tag{21}$$

This superposition enables the model to access all particle indices in parallel, laying the foundation for efficient quantum encoding and subsequent controlled operations.

3.2. Quantum state representation of particle features

For a particle flow consisting of $N = 2^n$ particles, we denote the particle set as $X = \{x_i \mid 0 \leqslant i < N, i \in \mathbb{Z}\}$, where each $x_i \in \mathbb{R}^d$ corresponds to a d-dimensional feature vector characterizing a single particle. To differentiate the features of individual particles in the quantum representation, the index register is initialized into a uniform superposition state. This register functions as a control mechanism in a controlled quantum circuit, facilitating the conditional application of quantum gates to encode particle-specific features.

The number of qubits required for feature encoding is determined by the chosen encoding strategy. Assuming m qubits are employed, the overall quantum state representing the particle ensemble is constructed by associating the features of each particle with a specific component of the superposition state in the index register. This controlled encoding process results in the following quantum structure:

$$|\psi_i\rangle_{\mathrm{i}=ir} = \mathrm{Ctrl}U^{\mathrm{ir}=i}(x_i)\,|0\rangle^{\otimes(n+m)} = \frac{1}{\sqrt{2^n}}|i\rangle^{\otimes n}\,U^{\mathrm{ir}=i}(x_i)\,|0\rangle^{\otimes m}. \tag{22}$$

In this framework, $|\psi_i\rangle_{\mathrm{i}=ir}$ denotes the quantum state encoding the features of the ith particle, while $U^{\mathrm{ir}=i}$ represents a unitary operation conditioned on the index register. Specifically, the control condition $\mathrm{ir} = i$ implies that the operation is activated only when the quantum state of the index register matches the binary representation of index i. Under this condition, the gate $U^{\mathrm{ir}=i}(x_i)$ is applied to encode the feature vector x_i of the ith particle. After applying the controlled operations across all index values, the resulting quantum state—encoded with m qubits per particle—is given by:

$$|\psi\rangle = \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle\,U^{\mathrm{ir}=i}(x_i)\,|0\rangle^{\otimes m}. \tag{23}$$

3.3. Quantum circuit implementation of the self-attention mechanism

Building on the discussion in section 2.2, particle features must be transformed to generate the corresponding queries, keys, and values required for the self-attention mechanism. In classical self-attention networks, this transformation is typically achieved through linear operations using randomly initialized, trainable matrices. In contrast, quantum self-attention networks utilize unitary operations to derive queries, keys, and values, embedding both the particle feature representations and trainable parameters within the quantum circuit.

Extending the quantum circuit architecture introduced in section 3.2, we incorporates trainable parameters into the unitary operations responsible for particle feature embedding. These parameterized operations are used to construct quantum representations of the queries, keys, and values.

Subsequently, the self-attention coefficients between particles are computed, which correspond to the inner products between the quantum-encoded queries and keys, as formulated in equation (18). In order to facilitate this computation, we design a quantum circuit that directly prepares quantum states whose amplitudes encode the desired self-attention coefficients. This approach, thereby, eliminates the need for direct measurement of individual particle features, enabling a coherent and efficient realization of quantum self-attention.

To illustrate the computation of the self-attention coefficient for the kth particle using m qubits, the query for the particle is first embedded via a unitary operation $U_Q(x_k, \theta_k)$. In contrast to the unitary gate $\mathrm{Ctrl}\,U^{\mathrm{ir}=i}$, which requires an index-conditioned control structure, this embedding is performed directly and independently, without relying on the index register. Here, $U_Q(x_k, \theta_k)$ denotes a trainable unitary matrix parameterized by the feature vector x_k and the associated parameter set θ_k, encoding the query state of the kth particle into a quantum representation,

$$|\varphi_{\mathrm{att}}\rangle = \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle\,U_Q(x_k,\theta_k)\,|0\rangle^{\otimes m}. \tag{24}$$

Subsequently, the self-attention coefficients between the kth particle and the remaining particles are constructed. Under different control conditions of the index qubits, bond embeddings of the particles are sequentially applied to the m qubits via parameterized unitary gates $[U_K^{\mathrm{ir}=i}(x_i, \theta_i)]^\dagger$. This process yields the complete set of key representations across all particles. The explicit structure o the unitary matrix is defined as follows:

$$|\varphi_{\text{att}}\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \left[U_K^{\text{ir}=i}(x_i,\theta_i)\right]^\dagger U_Q(x_k,\theta_k)\,|0\rangle^{\otimes m}. \tag{25}$$

As indicated in equation (20), the amplitude associated with the basis state $|0\rangle^{\otimes m}$ corresponds to the self-attention coefficient between particle features. To compute this amplitude explicitly, we expand the expression:

$$\left[U_K^{\text{ir}=i}(x_i,\theta_i)\right]^\dagger U_Q(x_k,\theta_k)\,|0\rangle^{\otimes m} = \gamma_{i0}|0\rangle^{\otimes m} + \sum_{j=1}^{2^m-1} \gamma_{ij}|j\rangle^{\otimes m}, \tag{26}$$

$$\begin{aligned} |\varphi_{\text{att}}\rangle &= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \left[U_K^{\text{ir}=i}(x_i,\theta_i)\right]^\dagger U_Q(x_k,\theta_k)\,|0\rangle^{\otimes m} \\ &= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \left(\gamma_{i0}|0\rangle^{\otimes m} + \sum_{j=1}^{2^m-1} \gamma_{ij}|j\rangle^{\otimes m}\right). \end{aligned} \tag{27}$$

Here, γ_{ij} denotes the amplitude of the quantum state corresponding to the ith particle embedded onto the jth quantum basis state among m qubits. Equation (27) formulates the quantum state in terms of self-attention coefficients between a given particle k and all other particles i. In particular, γ_{i0} represents the self-attention coefficient measuring the interaction between particle k and particle i. These coefficients can be interpreted as similarity scores computed between the query vector of particle k and the key vector of particle i.

Next, a weighted summation operation is applied to the value representations of particles within the self-attention mechanism. The self-attention coefficients—encoded in the amplitude distribution of the index register—serve as weighting factors in this operation. By reapplying the same control conditions of the index register, the model performs a weighted summation over the value embeddings of different particles to update the feature representation of each particle.

The number of qubits required for value embedding depends on the specific encoding strategy, with t qubits used to implement the value representation. Unitary gates $\text{Ctrl}\,U_V^{\text{ir}=i}(x_i,\theta_i)$ are employed to encode the particle values, ensuring consistency with the encoding process used for keys. When the quantum state of the index register satisfies the control condition $\text{ir} = i$, the value representation associated with particle x_i is embedded accordingly,

$$\begin{aligned} |\varphi_{\text{val}}\rangle &= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \left[U_K^{\text{ir}=i}(x_i,\theta_i)\right]^\dagger U_Q(x_k,\theta_k)\,|0\rangle^{\otimes m}\, U_V^{\text{ir}=i}(x_i,\theta_i)\,|0\rangle^{\otimes t} \\ &= \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \left(\gamma_{i0}|0\rangle^{\otimes m} + \sum_{j=1}^{2^m-1} \gamma_{ij}|j\rangle^{\otimes m}\right) |\varphi_{\text{val}=i}\rangle^{\otimes t}. \end{aligned} \tag{28}$$

Here, $|\varphi_{\text{val}=i}\rangle^{\otimes t}$ denotes the quantum state representing the value of the ith particle. A projection measurement onto the basis state $|0\rangle^{\otimes m}\langle 0|^{\otimes m}$ is performed on the self-attention register to extract the amplitude γ_{i0} corresponding to the $|0\rangle^{\otimes m}$ component. This amplitude serves as the self-attention coefficient, consistent with the attention mechanism.

The updated feature representation of each particle is then obtained by computing a weighted sum over the value states, with the extracted self-attention coefficients acting as weights. In the proposed design, these coefficients are encoded in the amplitude distribution of the index register. Since the self-attention register and the value register share the same index register, the post-measurement state of the self-attention register collapses to $|0\rangle^{\otimes m}$, ensuring that the self-attention coefficients remain encoded in the system amplitudes.

Consequently, the joint quantum state of the index and value registers encodes the weighted summation of particle values modulated by the corresponding self-attention coefficients. This operation is formally described using the density matrix formalism, where ρ denotes the pre-measurement density matrix of the composite quantum system,

$$\rho = |\varphi_{\text{val}}\rangle\langle\varphi_{\text{val}}|. \tag{29}$$

After applying the projection operator $I_{\text{ir}} \otimes |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes I_{\text{val}}$, the probability of obtaining the measurement outcome $|0\rangle^{\otimes m}$ on the self-attention register is given by:

$$P(0) = \text{Tr}\left[\left(I_{\text{ir}} \otimes |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes I_{\text{val}}\right)\rho\right]. \tag{30}$$

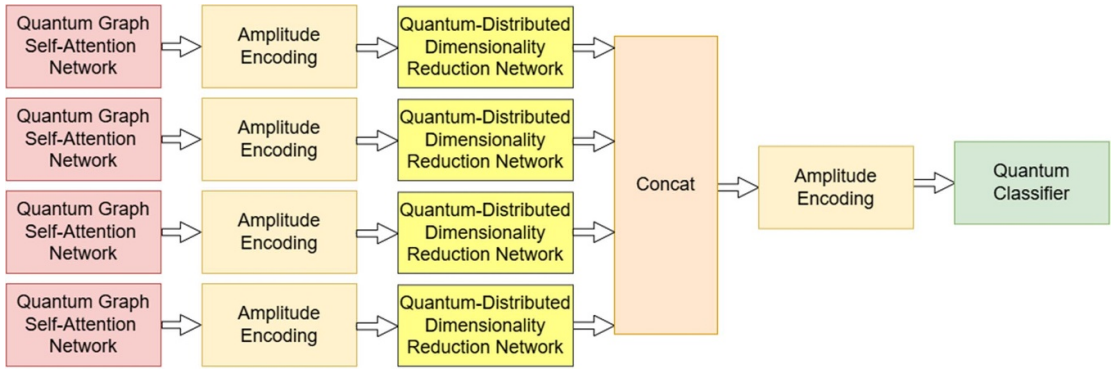

Figure 3. The overall architecture of the QCGAT system is shown, in which multiple quantum complete graph self-attention modules operate in parallel to extract features. The extracted features are then fused via a parallel quantum distributed dimensionality reduction network and subsequently fed into a quantum classifier.

After the measurement, the quantum system collapses to the following post-measurement state:

$$\rho_P = \frac{(I_{\text{ir}} \otimes |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes I_{\text{val}})\,\rho\,(I_{\text{ir}} \otimes |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes I_{\text{val}})}{P(0)}. \tag{31}$$

The joint density matrix of the index register and the value register, denoted by ρ_k, is expressed as:

$$\rho_k = \text{Tr}_m(\rho_P). \tag{32}$$

Here, Tr_m denotes the partial trace operation over the m qubits, resulting in a reduced quantum state corresponding to the joint subsystem of the index register and the value register, after performing an all-zero projection measurement on the self-attention register. This procedure enables the computation of self-attention coefficients among particle features, as well as the weighted aggregation of those features.

The resulting density matrix ρ_k is Hermitian. Given that all information in a Hermitian matrix can be fully determined by its main diagonal and either its upper or lower triangular part—owing to conjugate symmetry—we extract the upper triangular portion together with the diagonal entries as the output representation of the quantum module. This design choice accelerates the training of downstream tasks, reduces feature redundancy, and enhances overall model efficiency.

Although functionally equivalent to classical weighted summation, the implementation of QCGAT exhibits three intrinsic quantum-native characteristics.

First, it features a parallel and overlapping computation scheme, in which all attention coefficients are generated simultaneously during unitary evolution, thereby eliminating the need for pairwise calculations. Second, it employs amplitude encoding, where attention weights are inherently embedded in the amplitudes of quantum states, thereby avoiding cross-domain data transfer between classical and quantum systems. Third, it adopts deferred measurement, where normalization and aggregation are accomplished via post-selection performed at the end of computation, rather than through intermediate destructive measurements.

4. System model

The overall architecture of the proposed system is illustrated in figure 3. It consists of three primary components: the QCGAT module for particle representation learning, the quantum dimensionality reduction network, and the quantum classifier. The input particle flow features are first processed by QCGAT to extract enriched representations. Despite the compression of the density matrix by retaining only its upper triangular part, the resulting feature vector remains high-dimensional. To address this, we design a quantum distributed network to reduce the dimensionality of the feature vector. Finally, the reduced representation is passed to a quantum classifier built upon a quantum convolutional network to perform the classification task.

4.1. QCGAT module

To encode the particle features denoted by x_i, the quantum circuit illustrated below is employed. This work adopts angle encoding to transform particle features into quantum states.

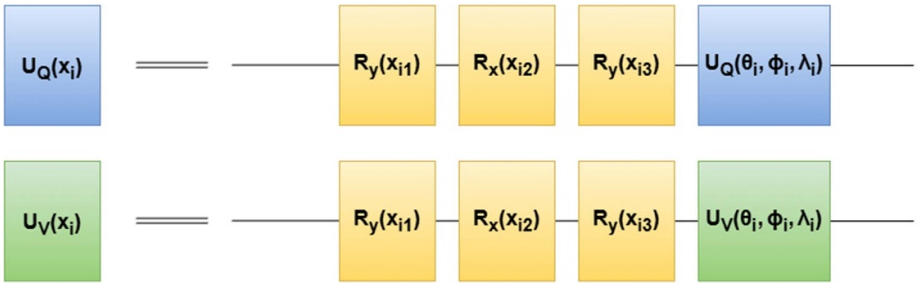

Figure 4. Quantum circuit for particle query and value encoding: after applying three sets of single-qubit rotation gates for angle encoding, trainable parameters are introduced to map the encoded features into the corresponding query and value representations.

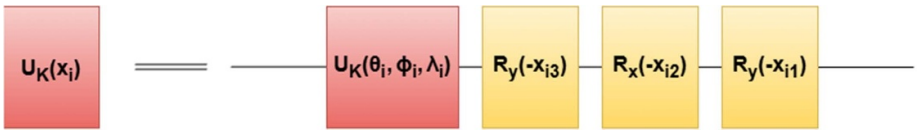

Figure 5. Quantum circuit for particle key encoding: constructed as the conjugate transpose of the query encoding circuit to maintain symmetry within the attention mechanism.

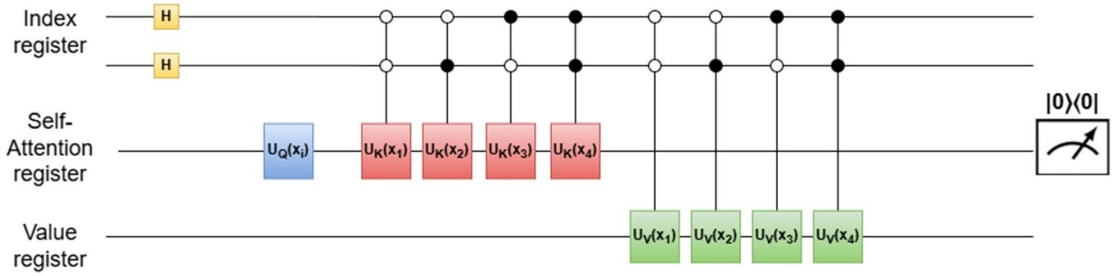

Figure 6. Quantum complete graph self-attention module: complete quantum circuit structure of the QCGAT module: the index register controls the self-attention and value registers to perform weighted summation of particle features and update the corresponding feature representations.

To enable learnable representations for queries, keys, and values, the single-qubit gate $U_3(\theta, \phi, \lambda)$ is employed, thereby introducing trainable parameters into the encoding process. The U_3 gate, which is widely used in quantum computing, is defined as follows:

$$U_3(\theta, \phi, \lambda) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}. \tag{33}$$

In the proposed encoding scheme, the parameter θ controls the rotation angle, determining the displacement of the quantum state on the Bloch sphere, while ϕ and λ serve as phase parameters, modulating the initial and final phases of the quantum state, respectively. The U_3 gate performs a rotation about the y-axis via θ, and about the z-axis via ϕ and λ. By tuning these three parameters, the U_3 gate is capable of implementing an arbitrary single-qubit unitary transformation.

To clearly distinguish the three trainable-parameter unitary gates, color coding is applied in all quantum circuit diagrams: query, key, and value correspond to blue, red, and green, respectively. Figure 4 illustrates the circuit architecture for embedding the particle queries and values. In this circuit, particle features are first encoded using three single-qubit rotation gates for angle encoding, after which the trainable parameters θ, ϕ, and λ are introduced to map the features to the corresponding query and value representations. Figure 5 presents the embedding circuit for the particle keys. As defined in equation (19), the key embedding circuit is the conjugate transpose of the query embedding circuit to maintain symmetry within the quantum attention mechanism. The complete structure of the QCGAT module is illustrated in figure 6.

In the experimental setup, particle features are first sorted in descending order based on their transverse momentum (p_T). The top four particles with the highest p_T values are selected to represent the

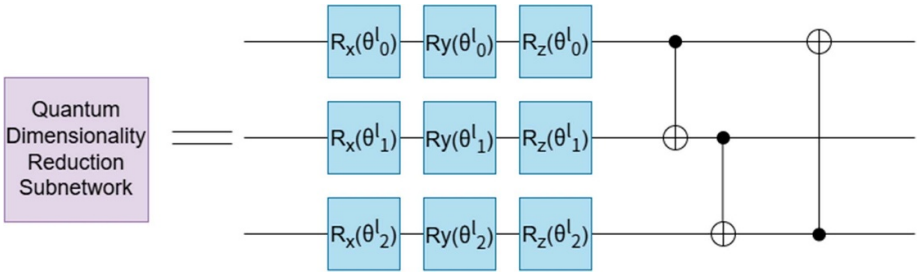

Figure 7. Quantum distributed dimensionality reduction subnetwork circuit structure: three layers of parallel rotation and controlled gates are employed to achieve the compression of high-dimensional particle features.

entire particle flow and are used in constructing a complete graph $G(4,6)$, where each node corresponds to a particle and each edge represents a potential interaction. Each node is associated with a particle feature vector of dimension 3. When accounting for self-connections required by the self-attention mechanism, the total number of edges increases to 10.

To implement the quantum self-attention mechanism, the circuit comprises three registers: the index register (requiring 2 qubits), the self-attention register (1 qubit), and the value register (1 qubit). Figure 6 depicts the quantum circuit used to compute the updated feature representation ρ_k for the kth particle. This process effectively realizes message passing and aggregation via the self-attention mechanism.

In this framework, the feature representation of each particle is updated in a uniform manner. As such, the quantum circuit illustrated in figure 6 is replicated four times—once for each particle—with the only variation being the query embedding for the kth particle in the self-attention register. Since each subcircuit operates on only 4 qubits, the computations for all particles can be executed in parallel, enabling efficient and scalable quantum feature updates.

4.2. Quantum distributed dimensionality reduction network

After processing the particle features using the QCGAT, the updated feature representations are obtained. Although the upper triangular elements of the density matrix are extracted to optimize the representation, the resulting feature vector remains high-dimensional. To facilitate downstream classification, dimensionality reduction is applied prior to feeding the features into the quantum classifier. To address this, we propose a quantum distributed network architecture that compresses the particle features into a lower-dimensional representation suitable for classification tasks. The structure of the proposed quantum dimensionality reduction subnetwork is illustrated in figure 7.

The quantum distributed dimensionality reduction network introduced herein demonstrates strong scalability and adaptability. Its architecture can be flexibly adjusted according to specific application requirements across several dimensions. First, the number of feature groups can be configured based on the dimensionality of the input data. Second, the number of qubits allocated per feature group can be optimized depending on the complexity of the features. Third, the number of features retained during feature fusion can be adjusted for the target task, which in turn influences the depth of the overall network. This multi-level configurability enables the network to accommodate data processing tasks of varying scales and characteristics. In the implementation presented in this work, each particle feature vector—expanded to 64 dimensions by the QCGAT module—is partitioned into eight groups, each containing eight features. Each group is encoded using three qubits via amplitude encoding. After passing through the first layer of the quantum dimensionality reduction subnetwork, the first two qubits are measured to produce the group-level output. Subsequently, the outputs of the first four groups and the last four groups are separately re-encoded using amplitude encoding with three qubits and fed into the next layer of the subnetwork. This process yields two group-level outputs, which are concatenated to form a final four-dimensional representation for each particle. The overall architecture of the quantum distributed dimensionality reduction network is illustrated in figure 8. After processing all four particle feature vectors, a 16-dimensional vector is produced, which is then fed into the downstream quantum classifier for final prediction.

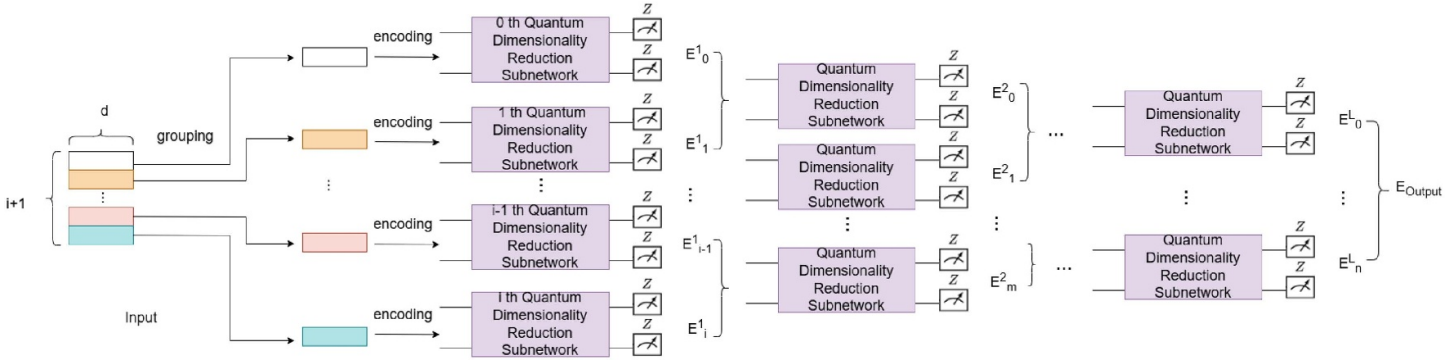

Figure 8. Overall architecture of the quantum distributed dimensionality reduction network: particle features are group-encoded, progressively compressed through multiple dimensionality-reduction layers, and fused into a low-dimensional representation.

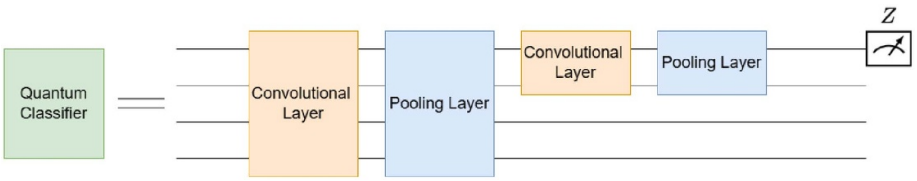

Figure 9. Overall architecture of the quantum classifier: multiple layers of quantum convolution and pooling operations progressively extract high-dimensional particle features, and the final classification probabilities are obtained through (Z)-basis measurement.

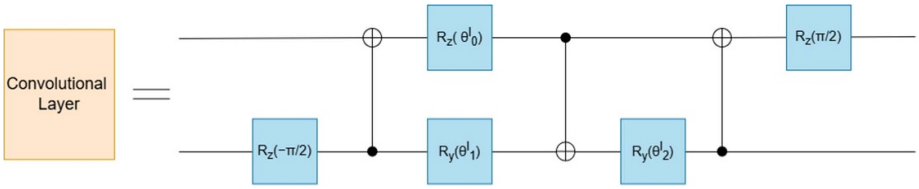

Figure 10. Quantum convolutional layer circuit structure: two-qubit unitary operations are alternately applied to even–odd index pairs with cyclic coupling to achieve feature extraction.

4.3. Quantum classifier

The QCNN is a prominent architecture in QML, recognized for its strong expressive capacity and reduced parameter complexity. The core principle of QCNNs lies in the use of quantum gate operations to realize convolution-like transformations, thereby facilitating effective feature extraction from quantum-encoded input data. Compared to their classical counterparts, QCNNs can achieve comparable representational power while utilizing significantly fewer trainable parameters. In a QCNN, convolution operations are implemented via parameterized quantum gates, which filter and transform input states in a localized fashion. These gates act analogously to classical convolution kernels by capturing relevant patterns or correlations present in the data. The overall network architecture is typically composed of alternating convolutional and pooling layers. The convolutional layers extract hierarchical features through entangling operations, whereas the pooling layers reduce the dimensionality of the quantum states, effectively lowering computational complexity and resource requirements

In this work, a QCNN is employed to perform the final classification of particle features. The complete architecture of the quantum classifier is illustrated in figure 9. The classification procedure is as follows: first, the four 4-dimensional particle feature vectors—produced by the quantum distributed dimensionality reduction network—are concatenated into a single 16-dimensional vector, thereby integrating the essential information from all particles. This 16-dimensional feature vector is then mapped onto a quantum state over 4 qubits using amplitude encoding. The encoded quantum state is subsequently processed through multiple layers of quantum convolution and pooling operations within the QCNN, progressively extracting and aggregating high-level representations of the particle features. The architectures of the quantum convolution and pooling layers are shown in figures 10 and 11, respectively. For quantum convolution, a two-qubit unitary operation is first applied to all even-indexed qubit pairs. This is followed by the same operation applied to all odd-indexed pairs, using a cyclic coupling scheme that connects both adjacent qubits and the first and last qubits through unitary gates. For quantum pooling, a two-qubit unitary gate is applied to each qubit pair. After the pooling operation, one qubit from each pair is retained for further computation, while the other is discarded. Finally, the remaining qubits are measured in the computational (Z) basis. The measurement outcomes are passed through a sigmoid activation function to produce the final classification probability of the particle flow.

This work adopts binary cross-entropy as the loss function L for binary classification tasks, with gradients computed via backpropagation during training,

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right)\log\left(1 - \hat{y}_i\right)\right]. \tag{34}$$

In this setting, y_i denotes the ground truth label (0 or 1), $\hat{y}_i$ denotes the probability predicted by the model, and N represents the total number of training samples. All experiments were conducted across five different random seeds to ensure statistical robustness of the results. Models were trained using the

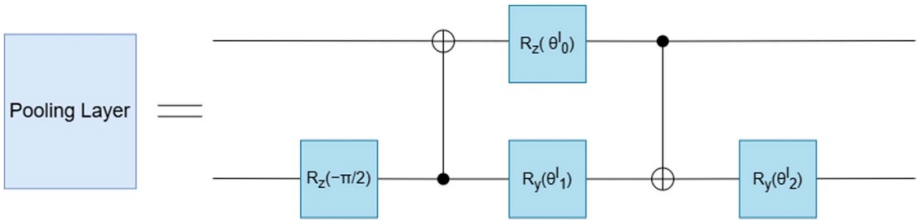

Figure 11. Quantum pooling layer circuit structure: after applying a unitary operation to each pair of qubits, one qubit is retained while the other is discarded to perform downsampling.

RAdam optimizer with a learning rate of 0.001, a batch size of 1024, and a maximum of 100 training epochs. To mitigate overfitting and enhance generalization, an early stopping strategy was employed based on the validation set AUC: training was terminated if no improvement (or a decline) in the validation AUC was observed over a predefined number of consecutive epochs. This mechanism substantially improved training efficiency and promoted more stable convergence. For rigorous and fair comparison, the quantum–classical hybrid and traditional quantum encoder baselines were trained under identical hyperparameters, optimizer settings, early stopping criteria, and random seed configurations. The experiments were implemented using the PennyLane [35] and PyTorch [36] libraries.

5. Results and analysis

5.1. Dataset

This section introduces the 'Top Quark Tagging dataset' [37], a simulated dataset designed for the development and evaluation of top quark tagging algorithms. The dataset comprises a total of 2 million events, partitioned into 1.2 million training events, 400 000 validation events, and 400 000 test events.

In this dataset, hadronically decaying top quarks are used as signal events, while QCD jets serve as the background. Each event contains a particle flow representation with transverse momentum (p_{Tjet}) in the range of 550–650 GeV. For labeling, signal events are assigned a value of 1, and background events (QCD jets) are labeled as 0. The dataset is pre-divided into training, validation, and test subsets, enabling streamlined experimentation. Its well-structured design and rich physics content make it an effective benchmark for assessing the performance of top quark tagging models based on both classical and QML methods.

This study conducts a series of experiments on the top quark tagging dataset to evaluate the performance of the proposed model in a binary classification setting. The particle features are preprocessed in accordance with the methodology described in [29]. Specifically, particle flows are clustered using the anti-k_{T} algorithm [38, 39], with a jet radius parameter of $R = 0.8$.

For each particle i, the input features include the transverse momentum fraction $z_i = p_{\mathrm{T}}/p_{\mathrm{Tjet}}$, the relative pseudorapidity $\Delta\eta_i = \eta_i - \eta_{\mathrm{jet}}$, and the relative azimuthal angle $\Delta\phi_i = \phi_i - \phi_{\mathrm{jet}}$. These features are subsequently subjected to a series of preprocessing steps to prepare the data for input into the model. The preprocessing transformations are defined as follows:

$$\begin{aligned} z_i &\rightarrow \tan^{-1}(z_i) \\ \Delta\eta_i &\rightarrow \frac{\pi}{2}\frac{\Delta\eta_i}{R} \\ \Delta\phi_i &\rightarrow \frac{\pi}{2}\frac{\Delta\phi_i}{R}. \end{aligned} \tag{35}$$

5.2. Comparison with classical classifier and traditional quantum encoders

The QCGAT model proposed in this study was trained on the TopQCD dataset, and the corresponding experimental results are summarized in table 1. To ensure a fair and comprehensive evaluation, QCGAT was benchmarked against both quantum–classical hybrid models and traditional quantum encoder baselines (figure 12) under comparable—or fewer—trainable parameter conditions.

For fairness, the quantum–classical hybrid model was designed to maintain a similar number of trainable parameters as QCGAT. In this setting, the fused particle representations produced by QCGAT were fed into a feed-forward neural network for classification. QCGAT achieves a test accuracy of 83.5%, representing a 1% improvement over the hybrid baseline (82.5%). QCGAT further attains an AUC of 88.7%, exceeding the hybrid model by more than 0.4%.

Table 1. Performance comparison of QCGAT with baseline models on the Top Quark Tagging dataset. Results are averaged over 5 random seeds with standard deviations. QCGAT achieves superior performance in both AUC and Accuracy, with its corresponding values highlighted in bold.

Model	Params	AUC	Accuracy
Baseline_R1	780	0.867 ± 0.003	0.810 ± 0.005
Baseline_R5	924	0.873 ± 0.003	0.819 ± 0.005
Baseline_R10	1104	0.876 ± 0.004	0.823 ± 0.004
Baseline_R15	1284	0.880 ± 0.002	0.828 ± 0.003
Baseline_R20	1464	0.874 ± 0.007	0.822 ± 0.010
Hybrid model	1165	0.883 ± 0.004	0.825 ± 0.003
QCGAT	**852**	$\mathbf{0.887 \pm 0.004}$	$\mathbf{0.835 \pm 0.005}$

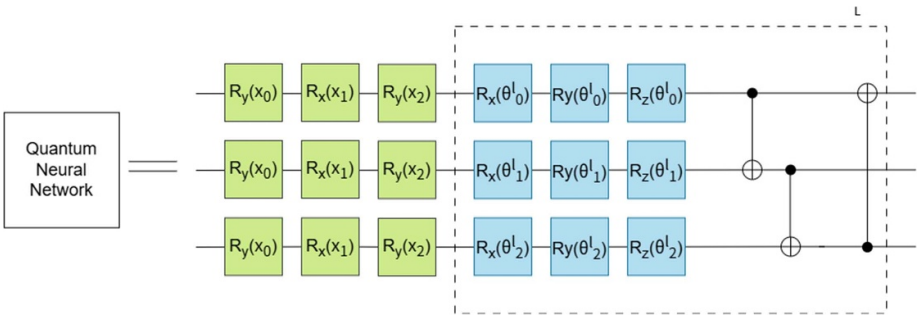

Figure 12. Baseline quantum encoder with a multi-layer variational architecture: particle features are directly angle-encoded and then passed through (L) stacked variational quantum layers for comparative analysis.

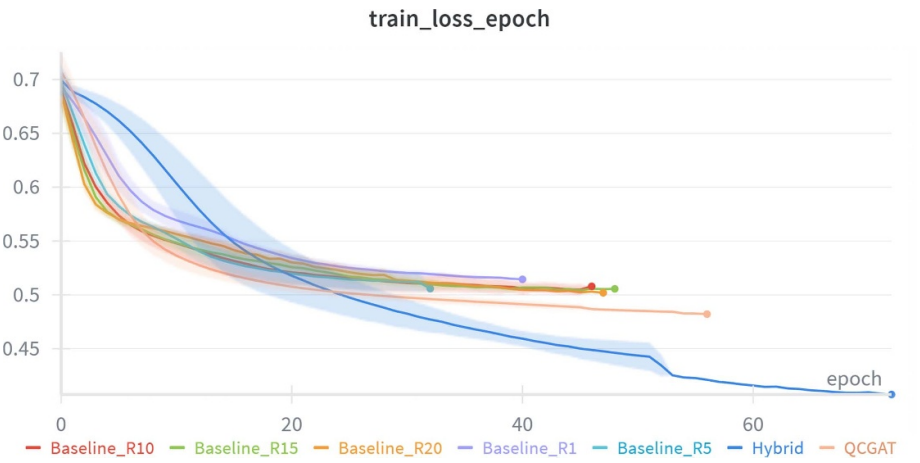

Figure 13. Training loss convergence comparison across different models: mean loss curves with standard error bands (shaded regions) computed over 5 random seeds, demonstrating the convergence performance of QCGAT compared to hybrid and traditional quantum encoder baselines.

The performance advantage becomes more pronounced when compared with conventional quantum encoders. The strongest baseline, tradition_R15, yields 82.8% accuracy, whereas QCGAT surpasses this result by 0.7% points while requiring fewer parameters. Moreover, relative to tradition_R5—which contains 924 parameters, comparable to the parameter count of QCGAT—the proposed model achieves a substantial 1.6% improvement in test accuracy, highlighting its superior parameter efficiency.

The training dynamics shown in figures 13–15 further support these observations. QGAT exhibits noticeably faster convergence, attains a lower final training loss, and demonstrates consistently narrower standard-deviation bands across multiple random seeds, indicating improved training stability. These results collectively show that the complete-graph quantum self-attention mechanism enhances the expressive capacity of the model and enables more effective capture of inter-particle dependencies, thereby outperforming both hybrid and conventional quantum encoder baselines.

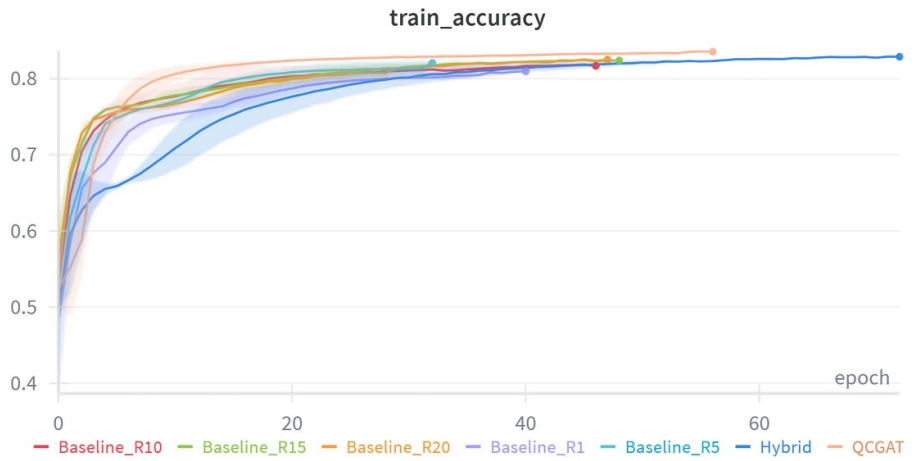

Figure 14. Training accuracy comparison across different models: mean accuracy curves with standard deviation bands (shaded regions) computed over 5 random seeds, demonstrating the learning performance and stability of QCGAT compared to hybrid and traditional quantum encoder baselines.

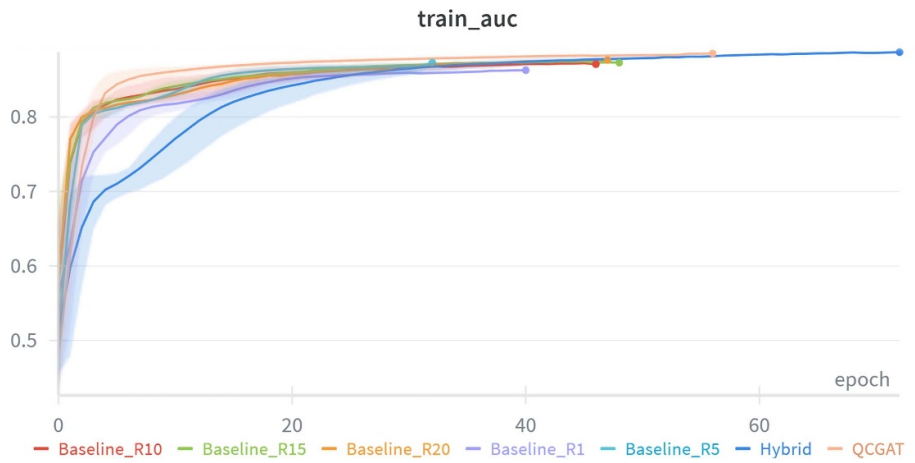

Figure 15. Training AUC comparison across different models: mean AUC curves with standard deviation bands (shaded regions) computed over 5 random seeds, demonstrating the discriminative performance and stability of QCGAT compared to hybrid and traditional quantum encoder baselines.

6. Conclusion

We present the QCGAT, a novel QML architecture specifically designed for particle flow classification. The proposed framework introduces a non-measurement quantum self-attention mechanism within an entangled quantum network and incorporates a quantum distributed dimensionality reduction structure that enables tunable cross-scale data adaptability. This design allows self-attention coefficients to be directly encoded into quantum state amplitudes, eliminating the need for quantum-to-classical information transfer while effectively capturing long-range dependencies among particles in high-energy physics events. We validate its effectiveness on the Top Quark Tagging dataset. Using only 852 trainable parameters, QCGAT achieves 83.5% accuracy and 88.7% AUC, outperforming the quantum–classical hybrid model by 1% in accuracy and 0.4% in AUC. Compared with the strongest traditional quantum encoder baseline, it improves accuracy by 0.7%; when compared with traditional quantum encoder baselines having comparable parameter counts, it achieves up to 1.6% higher accuracy. Moreover, QCGAT exhibits significantly faster convergence and superior training stability. Our approach imposes no restrictions on the underlying quantum circuit architecture and does not rely on any additional specialized training data. In fact, it can be seamlessly integrated into existing quantum GNNs without affecting their performance on standard classification tasks. When combined with variational optimization, the resulting model serves as a general-purpose framework for quantum-enhanced feature extraction.

In summary, QCGAT demonstrates the substantial potential of QML for high-energy physics, showing that a purely quantum self-attention mechanism based on complete-graph connectivity can surpass both quantum–classical hybrid models and traditional quantum encoder approaches on real-world HEP benchmarks while using fewer parameters. This work not only opens new avenues for quantum-accelerated particle physics analysis, but also suggests that highly expressive quantum attention mechanisms may play a pivotal role in the forthcoming era of fault-tolerant quantum computing.

Data availability statement

All data supporting the findings of this study are available within the article from the corresponding author Ting Li upon reasonable request. The data that support the findings of this study are available upon reasonable request from the authors https://zenodo.org/records/2603256.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grants Nos. 62271265.

ORCID iD

Shanglong Liu 0009-0003-7651-6388

References

[1] Brient J 2009 *J. Phys.: Conf. Ser.* **160** 012025
[2] Komiske P T, Metodiev E M and Thaler J 2019 *J. High Energy Phys.* **2019** 1–46
[3] Qu H and Gouskos L 2020 *Phys. Rev. D* **101** 056019
[4] Gilmer J, Schoenholz S S, Riley P F, Vinyals O and Dahl G E 2017 *Int. Conf. on Machine Learning* (PMLR) pp 1263–72 (available at: http://proceedings.mlr.press/v70/gilmer17a.html)
[5] Kipf T N and Welling M 2016 arXiv:1609.02907
[6] Velickovic P *et al* 2017 arXiv:1710.10903
[7] Abdughani M, Ren J, Wu L and Yang J M 2019 *J. High Energy Phys.* **2019** 55
[8] Arjona Martínez J, Cerri O, Spiropulu M, Vlimant J R and Vázquez M 2019 *Eur. Phys. J. Plus* **134** 333
[9] Ren J, Wu L and Yang J M 2020 *Phys. Lett. B* **802** 135198
[10] Mikuni V and Canelli F 2021 *Mach. Learn.: Sci. Technol.* **2** 035027
[11] Qu H, Li C and Qian S 2022 *Int. Conf. Machine Learning* (PMLR) pp 18281–92 (available at: https://icml.cc/virtual/2022/poster/17989)
[12] Cong I, Choi S and Lukin M D 2019 *Nat. Phys.* **15** 1273–8
[13] Smaldone A M, Kyro G W and Batista V S 2023 *Quantum Mach. Intell.* **5** 41
[14] Hu L *et al* 2019 *Sci. Adv.* **5** eaav2761
[15] Verdon G, McCourt T, Luzhnica E, Singh V, Leichenauer S and Hidary J 2019 arXiv:1909.12264
[16] Hu Z, Li J, Pan Z, Zhou S, Yang L, Ding C, Khan O, Geng T and Jiang W 2022 *2022 IEEE 40th Int. Conf. on Computer Design (ICCD)* (IEEE) pp 290–7
[17] Li G, Zhao X and Wang X 2024 *Sci. China Inf. Sci.* **67** 142501
[18] Chen F, Zhao Q, Feng L, Chen C, Lin Y and Lin J 2025 *Neural Netw.* **185** 107123
[19] Hoque S *et al* 2024 *Eur. Phys. J. C* **84** 1244
[20] Gianelle A, Koppenburg P, Lucchesi D, Nicotra D, Rodrigues E, Sestini L, de Vries J and Zuliani D 2022 *J. High Energy Phys.* **2022** 1–24
[21] Casals M *et al* 2025 *Mach. Learn.: Sci. Technol.* **6** 035048
[22] Liu Y, Arunachalam S and Temme K 2021 *Nat. Phys.* **17** 1013–7
[23] Muser T, Zapusek E, Belis V and Reiter F 2024 *Phys. Rev. A* **110** 032434
[24] Gyurik C and Dunjko V 2023 arXiv:2306.16028
[25] Belis V *et al* 2024 *Commun. Phys.* **7** 334
[26] Zhang S, Chen K-X and Yang J-C 2025 *Eur. Phys. J. C* **85** 378
[27] Pérez-Salinas A, Cervera-Lierta A, Gil-Fuster E and Latorre J I 2020 *Quantum* **4** 226
[28] Caro M C, Huang H Y, Cerezo M, Sharma K, Sornborger A, Cincio L and Coles P J 2022 *Nat. Commun.* **13** 4919
[29] Chen Y A and Chen K F 2025 *Phys. Rev. D* **111** 016020
[30] Weisstein E W Complete graph (available at https://mathworld.wolfram.com/CompleteGraph.html) from MathWorld–A Wolfram Web Resource (Accessed 11 February 2023)
[31] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L and Polosukhin I 2017 *Advances in Neural Information Processing Systems* vol 30 pp 5998–6008
[32] Elman J L 1990 *Cogn. Sci.* **14** 179–211
[33] LeCun Y, Bottou L, Bengio Y and Haffner P 1998 *Proc. IEEE* **86** 2278–324
[34] Markov V, Stefanski C, Rao A and Gonciulea C 2022 arXiv:2201.09845
[35] Team P 2024 Pennylane (available at: https://pennylane.ai)
[36] Team P 2024 Pytorch (available at: https://pytorch.org)
[37] Kasieczka G, Plehn T, Thompson J and Russel M 2019 Top quark tagging reference dataset *Zenodo* https://doi.org/10.5281/zenodo.2603256
[38] Cacciari M, Salam G P and Soyez G 2012 *Eur. Phys. J. C* **72** 1896
[39] Cacciari M, Salam G P and Soyez G 2008 *J. High Energy Phys.* **2008** 063