How Embeddings Shape Graph Neural Networks: Classical vs Quantum-Oriented Node Representations

Nouhaila Innan[1,2], Antonello Rosato[3], Alberto Marchisio[1,2], and Muhammad Shafique[1,2]
[1]eBRAIN Lab, Division of Engineering, New York University Abu Dhabi (NYUAD), Abu Dhabi, UAE
[2]Center for Quantum and Topological Systems (CQTS), NYUAD Research Institute, NYUAD, Abu Dhabi, UAE
[3]Department of Information Engineering, Electronics and Telecommunications, Sapienza University of Rome, Rome, Italy
{nouhaila.innan, alberto.marchisio, muhammad.shafique}@nyu.edu, antonello.rosato@uniroma1.it

***Abstract*—Node embeddings act as the information interface for graph neural networks, yet their empirical impact is often reported under mismatched backbones, splits, and training budgets. This paper provides a controlled benchmark of embedding choices for graph classification, comparing classical baselines with quantum-oriented node representations under a unified pipeline. We evaluate two classical baselines alongside quantum-oriented alternatives, including a circuit-defined variational embedding and quantum-inspired embeddings computed via graph operators and linear-algebraic constructions. All variants are trained and tested with the same backbone, stratified splits, identical optimization and early stopping, and consistent metrics. Experiments on five different TU datasets and on QM9 converted to classification via target binning show clear dataset dependence: quantum-oriented embeddings yield the most consistent gains on structure-driven benchmarks, while social graphs with limited node attributes remain well served by classical baselines. The study highlights practical trade-offs between inductive bias, trainability, and stability under a fixed training budget, and offers a reproducible reference point for selecting quantum-oriented embeddings in graph learning.**

***Index Terms*—Graph Neural Networks, Embeddings, Positional Encoding.**

I. Introduction

Graph-level prediction depends strongly on how node information is represented before and during message passing. Although node embeddings can substantially affect the information available to the graph classifier, their contribution is often difficult to assess in isolation because prior comparisons are frequently confounded by differences in backbone architectures, data splits, optimization settings, and training budgets.

This work studies, under a unified protocol, how different embedding modules influence downstream graph-level prediction when the GNN backbone is kept fixed. Specifically, we integrate a family of quantum-inspired embedding constructions into the same GNN pipeline and evaluate them on a shared set of graph classification benchmarks, including a molecular benchmark. Performance is measured using test Accuracy, Macro-F1, and Macro Precision/Recall. Rather than claiming that one embedding family is universally superior, the aim is to isolate the effect of the embedding stage and identify when different inductive biases are beneficial.

Across all experiments, each node is described using lightweight structural and positional information. Let x denote the base node feature vector, constructed from one-hot degree encodings, and let pe denote the Laplacian positional encoding (LPE) vector computed from the eigenvectors of the normalized graph Laplacian [1]. The notation $(x\|pe)$ denotes the concatenation of these two vectors. Some methods operate directly on this representation, while others generate additional features through graph-dependent dynamics and fuse them with the same base input.

The embedding variants considered in this paper cover a range of design principles. As classical controls, we include a **Fixed** embedding obtained through a fixed random projection of $(x\|pe)$ and a trainable multi-layer perceptron (**MLP**) applied to $(x\|pe)$. As a trainable quantum-circuit baseline, we include a variational quantum circuit (**VQC**) with angle encoding, denoted as **Angle-VQC**, where node features are encoded as circuit angles and measurement outcomes define node embeddings [2], [3]. We then consider three quantum-inspired constructions defined through graph dynamics:

- **QuOp**, which derives embeddings from local operator evolution on ego-neighborhood subgraphs,
- **QWalkVec**, which constructs node vectors from coined quantum-walk-style evolution over a fixed number of steps,
- **QPE**, which produces node descriptors from transition probabilities derived from a matrix-exponential formulation with anchor-node conditioning.

For both QuOp and QWalkVec, we evaluate non-trainable and trainable versions. In the results, the *trainable* variants are marked with a superscript *, indicating that a learnable fusion or projection stage is enabled.

These methods were selected to represent different ways of constructing node embeddings, including projection-based baselines, circuit-defined quantum embeddings, operator-based encodings, walk-based encodings, and spectral or positional constructions. Comparing them within one shared pipeline allows us to examine whether these different inductive biases lead to different graph-classification behavior under matched training conditions.

The central question of this study is: *given the same GNN backbone and training protocol, when does injecting operator- or walk-based structure into the embedding stage improve graph-level prediction relative to standard projection-based*

or MLP-based alternatives? To answer this, we vary only the embedding module while keeping the downstream architecture and training setup unchanged. We also compare trainable and non-trainable variants to distinguish gains arising from additional learnable capacity from those due to the embedding construction itself. Our working hypothesis is that embeddings encoding higher-order structure, walk dynamics, or positional information may be more effective when graph labels depend on multi-hop structural patterns, whereas simpler baselines may remain competitive when local descriptors are already sufficient.

The main contribution of this paper is a unified experimental framework for evaluating the role of the node embedding stage under a fixed GNN backbone and shared protocol. By reporting multiple metrics and, where applicable, both trainable and non-trainable configurations, the study clarifies when performance differences are attributable to the embedding design itself and when they are associated with additional trainable components. Because all methods are evaluated under the same finite optimization budget, the benchmark should be interpreted as a comparison under equal training resources rather than as an estimate of the maximum achievable performance of each method after exhaustive tuning.

The remainder of the paper presents the embedding constructions and their integration into the shared model, followed by an empirical analysis on TU [4] and QM9 [5] datasets, and a discussion of the methodological insights suggested by the observed performance patterns.

II. Background and Related Work

A. Node representations for graph classification

Graph neural networks (GNNs) rely on node-level inputs that determine what information can be propagated and pooled at the graph level [6]. In practical benchmarks, node attributes may be rich (molecular graphs) or entirely absent (social-network TU datasets such as IMDB variants). Graph benchmarking is a specific challenging problem [7], and when attributes are missing, a common strategy is to construct simple structural features (e.g., node degree) to provide a consistent input interface, while keeping the injected signal minimal [8]. In this work, we focus on how different node-embedding mechanisms, including quantum-oriented representations, affect downstream graph classification when the backbone and training protocol are held fixed.

B. Quantum-oriented node embeddings

Quantum-oriented embeddings fall into two broad families. The first family is *circuit-defined embeddings*, where a parameterized quantum circuit maps node features to a fixed-dimensional representation (e.g., Angle-VQC). The second family is *quantum-inspired embeddings*, which borrow primitives from quantum dynamics or quantum information (operators, walks, phase encodings) but are computed via graph operators and linear algebra [9]. Our benchmark includes three representative quantum-inspired methods from recent literature: QuOp [10], QWalkVec [11], and QPE [12]. While these approaches are motivated by quantum formalisms, in our implementation, they are evaluated under the same classical training pipeline and compared against classical baselines.

1) QuOp: quantum operator node representations: QuOp [10] constructs a node representation from local operator dynamics on an ego neighborhood. For each node, a neighborhood-induced operator is mapped to a unitary transformation, and summary statistics of the resulting state are used as the embedding. This yields a representation that explicitly encodes local graph structure while controlling dimensionality through a qubit budget. In our benchmark, we evaluate both a non-trainable variant (QuOp) and a trainable variant (QuOp*), where the trainable version introduces a learned projection that maps the operator-derived descriptors into the shared embedding dimension used by the downstream GIN (see Algorithm 1).

Algorithm 1 QuOp node embedding

Require: Graph $G = (V, E)$, node v, hop radius h, qubit budget q, output dim d
Ensure: Node embedding $\mathbf{z}_v \in \mathbb{R}^d$
1: Extract the h-hop ego-subgraph G_v centered at node v
2: Construct a graph-dependent matrix H_v from the local structure of G_v
3: Resize H_v to dimension $2^q \times 2^q$ to match the q-qubit state space
4: Compute the unitary matrix $U_v \leftarrow \exp(-i\tilde{H}_v)$
5: Initialize a basis state $|\psi_0\rangle$ associated with the target node
6: Evolve the state: $|\psi_v\rangle \leftarrow U_v|\psi_0\rangle$
7: Extract a fixed summary vector $\mathbf{s}_v$ from the evolved state $|\psi_v\rangle$
8: Map $\mathbf{s}_v$ to the shared embedding dimension: $\mathbf{z}_v \leftarrow \textsc{Proj}(\mathbf{s}_v)$ ▷ identity for QuOp; learnable for QuOp*
9: **return** $\mathbf{z}_v$

2) QWalkVec: node embeddings from quantum-walk dynamics: QWalkVec [11] constructs node embeddings by simulating a coined quantum walk and recording how node visitation probabilities evolve over time. In this way, each node is represented by a temporal descriptor that reflects how information propagates through multi-hop graph structure under the walk dynamics. We benchmark a fixed variant (QWalkVec) and a trainable variant (QWalkVec*), where the latter learns a projection from these time-series probability descriptors to the shared embedding dimension used downstream. Empirically, this projection can be important for aligning the walk-derived signal with the supervised task (see Algorithm 2).

3) QPE: quantum positional encodings for graphs: QPE [12] injects positional information into node representations through quantum-inspired phase dynamics. The main idea is to encode node position through the spectral structure of a graph-derived operator, such as a Laplacian-based matrix, and to evaluate the time evolution $U(t) = \exp(-iHt)$ at multiple time points. This produces node descriptors that reflect how each node responds to the same operator dynamics relative to a selected set of anchor nodes. In our benchmark, QPE is implemented as a quantum-inspired linear-algebra construction and evaluated under the same fixed GIN backbone (see Algorithm 3).

III. Methodology

Our goal is to quantify how the *node embedding stage* affects graph-level prediction under an otherwise fixed pipeline. Let

Algorithm 2 QWalkVec

Require: Graph $G = (V, E)$, steps T, walk parameters (w_p, w_q), output dim d
Ensure: Node embeddings $\{\mathbf{z}_v\}_{v\in V}$, $\mathbf{z}_v \in \mathbb{R}^d$
Construct the directed-edge state space $\mathcal{S} \leftarrow \{(u \to v) : (u, v) \in E\}$
Initialize the walk state $|\Psi^{(0)}\rangle$ on $\mathcal{S}$
for $t = 1$ **to**T **do**
    Apply the coin and shift updates:
        $|\Psi^{(t)}\rangle \leftarrow S\,C(w_p, w_q)\,|\Psi^{(t-1)}\rangle$
    For each node v, compute the node visitation probability $p_v^{(t)}$ by summing the probabilities of all directed-edge states in $|\Psi^{(t)}\rangle$ that end at or are associated with v
end for
for each node $v \in V$ **do**
    Form the time-series descriptor $\mathbf{s}_v \leftarrow [p_v^{(1)}, \ldots, p_v^{(T)}]$
    Map $\mathbf{s}_v$ to the shared embedding dimension: $\mathbf{z}_v \leftarrow \textsc{Proj}(\mathbf{s}_v)$ ▷ fixed for QWalkVec; learnable for QWalkVec*
end for
return $\{\mathbf{z}_v\}_{v\in V}$

Algorithm 3 QPE

Require: Graph $G = (V, E)$, operator H, time set $\mathcal{T}$, number of anchors A, output dim d
Ensure: Node embeddings $\{\mathbf{z}_v\}_{v\in V}$, $\mathbf{z}_v \in \mathbb{R}^d$
Select a set of anchor nodes $\mathcal{A} \subseteq V$ with $|\mathcal{A}| = A$
Compute the spectral decomposition $H = V\Lambda V^\top$
for each $t \in \mathcal{T}$ **do**
    Compute the evolution operator $U(t) \leftarrow V \exp(-i\Lambda t)V^\top$
    for each anchor $a \in \mathcal{A}$ **do**
        Define the anchor indicator vector $\mathbf{e}_a$
        Evolve the anchor state: $\mathbf{u}_{a,t} \leftarrow U(t)\mathbf{e}_a$
        for each node $v \in V$ **do**
            Append $\mathbf{u}_{a,t}[v]$ to the descriptor list of node v
        end for
    end for
end for
for each node $v \in V$ **do**
    Concatenate all collected values into a descriptor vector $\mathbf{s}_v$
    Map $\mathbf{s}_v$ to the shared embedding dimension: $\mathbf{z}_v \leftarrow \textsc{Proj}(\mathbf{s}_v)$ ▷ fixed in our QPE configuration
end for
return $\{\mathbf{z}_v\}_{v\in V}$

$G = (V, E)$ denote an input graph with $|V|$ nodes and edge set E. Let ϕ_θ denote the node-embedding module producing $\mathbf{Z} \in \mathbb{R}^{|V|\times d}$.

The resulting node embeddings are then consumed by the same downstream classifier $f_\psi(\cdot)$ (a fixed GIN backbone) to produce graph logits and predictions. As shown in Fig. 1, across all runs, we keep constant: (i) the backbone architecture f_ψ, (ii) the data splits, (iii) the optimizer and early-stopping criterion, and (iv) the evaluation metrics. Therefore, differences in results are attributable to ϕ_θ (and to whether ϕ_θ is trainable), rather than to confounding changes in message passing or the training procedure.

A. Node embedding constructions

Each embedding method defines a mapping from the graph and per-node descriptors to a fixed-dimensional node representation,

$$\mathbf{Z} = \phi_\theta(G, \{\mathbf{u}_v\}_{v\in V}) \in \mathbb{R}^{|V|\times d}, \tag{1}$$

where d_u denotes the dimensionality of the input node descriptor $\mathbf{u}_v$, and $\mathbf{z}_v$ denotes the v-th row of $\mathbf{Z}$. For all methods, the embedding stage can be expressed as the composition of a method-specific descriptor and a mapping into the shared space:

$$\mathbf{s}_v = g(G, v; \alpha) \in \mathbb{R}^{d_s}, \tag{2}$$

$$\mathbf{z}_v = \rho_\theta([\mathbf{u}_v \| \mathbf{s}_v]) \in \mathbb{R}^d. \tag{3}$$

The function $g(\cdot)$ is determined by the chosen embedding construction and fixed hyperparameters α (e.g., walk horizon, anchor count, or qubit budget). The map $\rho_\theta(\cdot)$ projects the resulting descriptor into the shared embedding dimension d. In the non-trainable setting, ρ_θ is fixed; in the trainable setting, ρ_θ contains learnable parameters. The **Trainable** indicator used in the results indicates whether the embedding stage introduces learnable parameters beyond the shared GIN backbone (e.g., a learnable projection/fusion map and, for Angle-VQC, circuit parameters). Detailed algorithmic descriptions of the quantum-inspired embeddings (QuOp, QWalkVec, and QPE) are provided in Sec. II.

a) Classical baselines: Classical controls operate directly on $\mathbf{u}_v$ by setting $\mathbf{s}_v = \varnothing$ in Eq. (3). The fixed baseline applies a fixed random projection,

$$\mathbf{z}_v = \mathbf{W}_0\,\mathbf{u}_v, \qquad \mathbf{W}_0 \in \mathbb{R}^{d\times d_u} \text{ fixed}, \tag{4}$$

whereas the MLP baseline replaces the fixed map with a trainable feed-forward transformation,

$$\mathbf{z}_v = \mathrm{MLP}_\theta(\mathbf{u}_v). \tag{5}$$

b) Circuit-defined embedding: Angle-VQC generates per-node descriptors by applying a parameterized quantum circuit to an encoding of $\mathbf{u}_v$. First, $\mathbf{u}_v$ is mapped to q rotation angles,

$$\boldsymbol{\varphi}_v = \mathbf{A}\mathbf{u}_v \in \mathbb{R}^q, \tag{6}$$

followed by an angle-encoding layer and L_q variational layers (see Fig. 2), yielding the state

$$|\psi_v\rangle = U(\boldsymbol{\varphi}_v, \boldsymbol{\theta})\,|0\rangle^{\otimes q}. \tag{7}$$

A real-valued circuit descriptor is obtained by measuring a fixed observable set $\{\hat{O}_k\}_{k=1}^m$. In our experiments, we use single-qubit Pauli-Z measurements on each qubit, i.e., $m = q$ and $\hat{O}_k = Z_k$:

$$s_{v,k} = \langle\psi_v|\hat{O}_k|\psi_v\rangle, \qquad k = 1, \ldots, m. \tag{8}$$

This defines $\mathbf{s}_v = [s_{v,1}, \ldots, s_{v,m}] \in \mathbb{R}^m$, which is mapped to the shared node embedding dimension via Eq. (3).

B. Graph classifier and training protocol

All embedding variants are evaluated with the same downstream classifier f_ψ implemented as a GIN backbone [13]. Starting from $\mathbf{H}^{(0)} = \mathbf{Z}$, message passing produces node states after L layers:

$$\mathbf{H}^{(\ell)} = \mathrm{GIN}^{(\ell)}(\mathbf{H}^{(\ell-1)}, G), \qquad \ell = 1, \ldots, L, \tag{9}$$

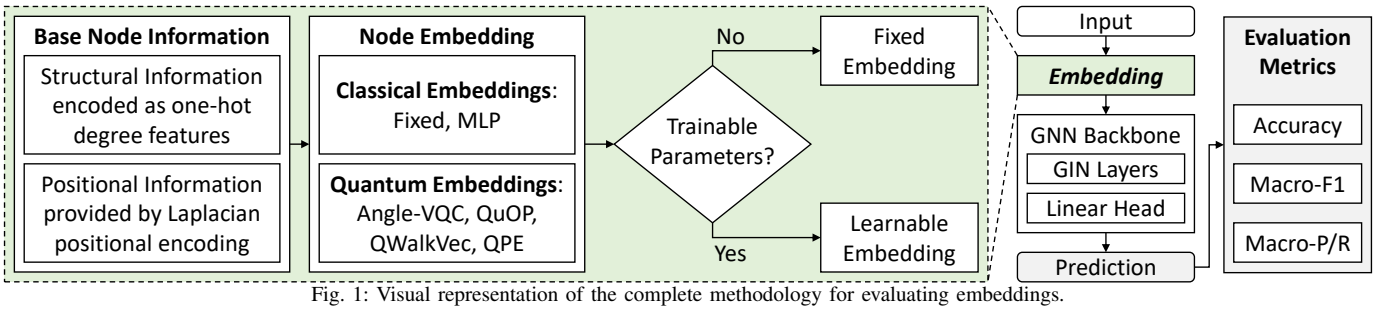

Fig. 1: Visual representation of the complete methodology for evaluating embeddings.

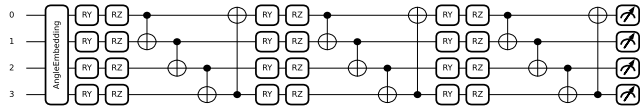

Fig. 2: VQC used as the quantum embedding module. Angle embedding (Y rotations) is followed by L_q entangling layers, each composed of per-qubit R_Y/R_Z rotations and a ring of CNOT gates. Measurements include single-qubit Z expectations.

followed by a permutation-invariant readout and linear head:

$$\mathbf{h}_G = \mathrm{Pool}(\mathbf{H}^{(L)}), \qquad \hat{\mathbf{y}}_G = \mathbf{W}_c \mathbf{h}_G + \mathbf{b}_c. \tag{10}$$

For a dataset $\mathcal{D} = \{(G_i, y_i)\}_{i=1}^{N}$, parameters are optimized by minimizing cross-entropy over the training split,

$$\min_{\theta,\psi} \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(G,y)\in\mathcal{D}_{\text{train}}} \mathcal{L}\big(\hat{\mathbf{y}}_G, y\big), \tag{11}$$

with early stopping based on validation Macro-F1 and reporting of test Accuracy, Macro-F1, and Macro Precision/Recall. Across all runs, the backbone f_ψ, stratified splits, optimization settings, early-stopping rule, and metric computation are kept fixed; consequently, performance differences reflect the embedding construction and its trainability.

IV. Results and Discussion

A. Experimental Setup

We evaluate all embedding variants within a controlled graph-classification pipeline to isolate representation effects. All methods share the same downstream classifier (GIN backbone), identical stratified splits, identical optimization settings, and the same early-stopping criterion. The comparison includes two classical baselines (Fixed and MLP), avoiding more complex systems such as Transformers [14], and a set of quantum-oriented embeddings. Among the quantum-oriented variants, Angle-VQC is the only circuit-defined embedding (node representations are generated by a parameterized quantum circuit using PennyLane's `default.qubit` simulator [15]), whereas QuOp, QWalkVec, and QPE are quantum-inspired constructions computed from graph operators and linear-algebraic transformations. Under this fixed protocol, performance differences primarily reflect the quality of the node representations produced by each embedding. The details for the evaluation setup are reported in Table I.

Table II reports test-set Accuracy, Macro-F1, and Macro-P/R for all embedding variants under this protocol, and the following analysis discusses the observed dataset-dependent trends and the conditions under which quantum-oriented embeddings provide improvements over strong classical baselines.

TABLE I: Hyperparameters and evaluation protocol used for all embedding variants (unless stated otherwise).

Category	Setting
Datasets	TU: IMDB-BINARY, IMDB-MULTI, MUTAG, PROTEINS, ENZYMES; QM9 (classification via binning).
Splits	Stratified 80/10/10 (train/val/test), seed $= 7$.
Positional encoding	Laplacian eigenvector PE, $k = 8$, stored as `pe`.
Embedding output	Node embedding dim $d = 32$ (all methods).
Backbone	GIN (3 layers), hidden dim 64, global mean pooling; MLP head $64 \to 64 \to C$; dropout 0.2.
Optimization	Adam; lr 10^{-3}; weight decay 0; max epochs 30.
Early stopping	Monitor val Macro-F1; patience 7; select best checkpoint.
Batch size	16.
QM9 labeling	Target index 0; 2 bins; quantile binning; max graphs 5000.
QWalkVec	$t = 32$, $w_p = 0.5$, $w_q = 4.0$; stored as `qwalkvec`.
QPE	Times $\{0.5, 1.0, 2.0\}$; anchors 8

B. Dataset-wise performance patterns

Performance varies systematically across dataset families. On **IMDB**, the strongest results are obtained by the classical baselines (MLP: Acc $= 0.72$, Macro-F1 $= 0.7172$; Fixed: Acc $= 0.71$, Macro-F1 $= 0.7097$). In this attribute-scarce regime, where node features are constructed to ensure $\mathbf{x} \neq \varnothing$ but remain intentionally minimal, additional embedding complexity does not consistently translate into improved class-balanced performance. Angle-VQC underperforms sharply (Acc $= 0.60$, Macro-F1 $= 0.4667$) and exhibits a pronounced Macro-P/R mismatch, indicating uneven class-wise behavior under the current training budget. QuOp/QWalkVec/QPE remain competitive but do not surpass the strongest baselines on IMDB.

On **IMDB-MULTI**, the ranking changes and quantum-inspired structures become beneficial. QuOp achieves the best overall result (Acc $= 0.5067$, Macro-F1 $= 0.4662$), improving over Fixed/MLP (Acc ≈ 0.43–0.44, Macro-F1 ≈ 0.39–0.41). The trainable QuOp* does not exceed QuOp (similar Macro-F1 but lower accuracy), suggesting that the operator-induced inductive bias is effective on its own, while additional trainable capacity is not consistently converted into better generalization in this setting. Angle-VQC attains moderate accuracy but does not improve Macro-F1 relative to QuOp, consistent with non-uniform class performance in the multi-class regime.

For structure-driven benchmarks, quantum-oriented embeddings yield the clearest gains over baselines.

On **MUTAG**, QWalkVec* dominates with Acc $= 0.9474$ and Macro-F1 $= 0.9360$, clearly exceeding all alternatives; Angle-

TABLE II: Performance comparison across embedding variants (test set). Best values are highlighted per dataset (Acc and Macro-F1).

Data	Method	Tr.	Acc	F1	P/R
IMDB	Fixed	N	0.71	0.7097	0.7237/0.7202
	MLP	Y	**0.72**	**0.7172**	0.7172/0.7172
	Angle-VQC	Y	0.60	0.4667	0.7895/0.5556
	QuOp	N	0.62	0.6200	0.6263/0.6263
	QuOp*	Y	0.65	0.6491	0.6500/0.6515
	QWalkVec	N	0.67	0.6576	0.7520/0.6959
	QWalkVec*	Y	0.68	0.6716	0.6776/0.6707
	QPE	N	0.63	0.6297	0.6322/0.6333
IMDB-MULTI	Fixed	N	0.4267	0.3914	0.4033/0.4267
	MLP	Y	0.4400	0.4132	0.4232/0.4400
	Angle-VQC	Y	0.4467	0.4024	0.4115/0.4467
	QuOp	N	**0.5067**	**0.4662**	0.5280/0.5067
	QuOp*	Y	0.4733	0.4660	0.4703/0.4733
	QWalkVec	N	0.4267	0.4299	0.4361/0.4267
	QWalkVec*	Y	0.4533	0.4244	0.4370/0.4533
	QPE	N	0.4200	0.4058	0.4028/0.4200
MUTAG	Fixed	N	0.7895	0.7841	0.8000/0.8462
	MLP	Y	0.8421	0.8348	0.8333/0.8846
	Angle-VQC	Y	0.8947	0.8782	0.8782/0.8782
	QuOp	N	0.8421	0.8348	0.8333/0.8846
	QuOp*	Y	0.8421	0.8348	0.8333/0.8846
	QWalkVec	N	0.4211	0.3942	0.6765/0.5769
	QWalkVec*	Y	**0.9474**	**0.9360**	0.9643/0.9167
	QPE	N	0.8421	0.7816	0.9062/0.7500
PROTEINS	Fixed	N	0.7679	0.7617	0.7596/0.7658
	MLP	Y	0.7589	0.7360	0.7606/0.7292
	Angle-VQC	Y	0.6786	0.5714	0.7813/0.6036
	QuOp	N	0.7054	0.6647	0.7083/0.6625
	QuOp*	Y	0.6518	0.6365	0.6370/0.6360
	QWalkVec	N	0.7321	0.7114	0.7247/0.7068
	QWalkVec*	Y	**0.7768**	**0.7630**	0.7708/0.7587
	QPE	N	0.7321	0.6973	0.7408/0.6922
ENZYMES	Fixed	N	0.2667	0.2160	0.1981/0.2667
	MLP	Y	0.2500	0.2242	0.2177/0.2500
	Angle-VQC	Y	0.1500	0.1317	0.1402/0.1500
	QuOp	N	0.1833	0.1871	0.2026/0.1833
	QuOp*	Y	0.1833	0.1658	0.1703/0.1833
	QWalkVec	N	**0.3000**	**0.2827**	0.2805/0.3000
	QWalkVec*	Y	0.1833	0.1556	0.1627/0.1833
	QPE	N	0.2500	0.2066	0.1793/0.2500
QM9	Fixed	N	0.8300	0.8300	0.8300/0.8300
	MLP	Y	0.8120	0.8107	0.8211/0.8120
	Angle-VQC	Y	0.8060	0.8053	0.8102/0.8060
	QuOp	N	0.7240	0.7152	0.7557/0.7240
	QuOp*	Y	0.7520	0.7502	0.7593/0.7520
	QWalkVec	N	0.7200	0.7162	0.7325/0.7200
	QWalkVec*	Y	**0.8520**	**0.8518**	0.8543/0.8520
	QPE	N	0.7780	0.7746	0.7956/0.7780

VQC is also strong (Acc $= 0.8947$, Macro-F1 $= 0.8782$), while Fixed/MLP/QuOp* cluster around Acc ≈ 0.84. The contrast within QWalkVec is informative: the non-trainable variant collapses (Acc $= 0.4211$) while QWalkVec* excels, indicating that walk-derived descriptors require a task-aligned projection to become effective.

On **PROTEINS**, QWalkVec* yields highest Macro-F1 (Acc $= 0.7768$, Macro-F1 $= 0.7630$), providing a modest improvement over Fixed (Acc $= 0.7679$, Macro-F1 $= 0.7617$).

On **ENZYMES**, absolute scores remain low across all methods, but QWalkVec achieves the strongest result among evaluated variants (Acc $= 0.3000$, Macro-F1 $= 0.2827$), improving over Fixed (Acc $= 0.2667$, Macro-F1 $= 0.2160$), indicating that richer multi-hop structural descriptors are still useful in a harder multi-class setting.

Finally, on **QM9** (binary classification via target binning), QWalkVec* again provides the best performance (Acc $= 0.8520$, Macro-F1 $= 0.8518$), improving over Fixed (Acc $= 0.8300$), while QuOp* and QPE are weaker in this configuration.

C. Where the quantum-oriented benefit comes from

Two factors explain when quantum-oriented embeddings improve over baselines. First, the advantage is strongest when labels correlate with *multi-hop structure* rather than weak constructed node attributes, as reflected by the consistent gains of QWalkVec* on MUTAG and QM9 and its improvements on PROTEINS and ENZYMES. Second, the effect depends on how the quantum-oriented signal is integrated into the representation space. QWalkVec shows that a learned projection can be essential: without it, walk-derived descriptors may be misaligned with the downstream loss (MUTAG collapse), whereas with a trainable mapping, the same structural signal becomes highly predictive. In contrast, QuOp illustrates that a fixed operator-based inductive bias can already be effective on IMDB-MULTI, while adding trainable capacity (QuOp*) does not reliably improve outcomes under a fixed optimization budget.

D. Hard regimes and failure modes suggested by ENZYMES.

ENZYMES stands out as uniformly difficult under the present configuration: all methods obtain low Macro-F1 and Accuracy (best Acc. $= 0.30$). Given that the backbone and training protocol are held fixed, this suggests that the tested embeddings, under the current backbone capacity and training budget, are not sufficient to elicit strong performance on this multi-class setting. Methodologically, this motivates targeted ablations in regimes where gains are modest, for example, varying the embedding fusion strategy, revisiting the scaling between base inputs and dynamics-derived features, or increasing shared-backbone capacity while maintaining the same controlled comparison framework.

E. Computational characteristics implied by the implementations.

The embedding families differ substantially in their computational profile as implemented here, which matters when interpreting practical trade-offs. QuOp requires per-node construction of an ego-neighborhood, padding to a power-of-two dimension, and a matrix exponential to obtain a unitary before measurement-style summaries; the cost therefore scales with both neighborhood size and the chosen qubit budget. QPE requires a per-graph eigendecomposition to evaluate $U(t) = \exp(-iHt)$, after which multiple time points can be evaluated cheaply but still depend on dense linear algebra on the graph size. QWalkVec builds a directed-edge state space and iterates a coined-walk update for a fixed number

of steps, accumulating node probabilities over time; its cost is dominated by the walk horizon and the number of directed edges. These differences motivated a conservative batch size and early stopping strategy in the shared training protocol, ensuring that comparisons remain feasible across datasets while keeping the downstream model fixed.

F. Trainability and stability

Trainability is not uniformly beneficial and should be treated as a controlled design axis. QWalkVec exhibits the strongest dependence on trainability (essential on MUTAG and beneficial on PROTEINS/QM9), whereas QuOp* shows that frozen representations can be preferable (IMDB-MULTI). Angle-VQC further illustrates dataset sensitivity [16]: it performs strongly on MUTAG but fails to generalize on IMDB and degrades on PROTEINS, indicating that circuit-defined embeddings can be more sensitive to the data regime and to optimization constraints.

G. Discussion and takeaways

Our results support three takeaways for embedding selection in graph classification.

- **Quantum-oriented embeddings can yield clear gains** in structure-driven regimes: walk-based quantum-inspired embeddings (QWalkVec*) provide the best performance on MUTAG and QM9 and remain competitive or best on PROTEINS and ENZYMES, demonstrating that quantum-motivated structural descriptors can outperform strong baselines under a fixed backbone.
- **The advantage is dataset-dependent**: on attribute-scarce social graphs (IMDB), classical baselines remain strongest, suggesting that the limiting factor is the weakness of node-level signal rather than embedding expressiveness.
- **Optimization stability matters as much as inductive bias**: trainability can be essential (QWalkVec) or neutral/negative (QuOp on IMDB-MULTI), so fair comparisons should report both frozen and trainable variants and rely on Macro-F1/Macro-P/R to expose class-wise failure modes that accuracy alone can mask.

V. Conclusion

This study isolates the effect of the node-embedding stage in graph classification by evaluating classical and quantum-oriented node representations under a fixed backbone (GIN), fixed splits, and a shared optimization protocol. The results show a consistent pattern: embeddings that explicitly inject multi-hop structural information can improve graph-level performance when labels are structure-driven, while attribute-scarce social graphs remain well served by simple baselines built from lightweight structural and positional cues.

Among the quantum-oriented variants, walk-derived descriptors are the most reliable source of gains, but only when their outputs are mapped into a task-aligned representation space: the strongest improvements appear when the walk signal is coupled to a learnable projection, whereas the corresponding frozen configuration can underperform sharply. Operator-based encodings exhibit a different profile, providing competitive performance without always benefiting from additional trainable capacity, which suggests that the inductive bias of the construction itself can be the dominant factor under a fixed training budget. Circuit-defined embeddings show clear dataset sensitivity, indicating that trainability and optimization stability remain practical constraints when the embedding is produced by a variational circuit.

Overall, the benchmark supports a simple takeaway for practitioners: when the target depends on graph structure beyond immediate neighborhoods, quantum-inspired dynamics can be a useful way to enrich node representations, but their benefit depends on how the resulting descriptors are interfaced with the downstream learner. A natural next step is to stress-test these findings under larger training budgets and multi-seed evaluation, and to study whether the same embedding choices remain favorable when paired with backbones that are explicitly designed for long-range dependencies.

Acknowledgment

This work was supported in part by the NYUAD Center for Quantum and Topological Systems (CQTS), funded by Tamkeen under the NYUAD Research Institute grant CG008.

References

[1] V. P. Dwivedi *et al.*, "Graph neural networks with learnable structural and positional representations," *arXiv preprint arXiv:2110.07875*, 2021.

[2] M. Cerezo *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, 2021.

[3] N. Innan *et al.*, "Financial fraud detection using quantum graph neural networks," *Quantum Machine Intelligence*, vol. 6, no. 1, p. 7, 2024.

[4] C. Morris *et al.*, "Tudataset: A collection of benchmark datasets for learning with graphs," *arXiv preprint arXiv:2007.08663*, 2020.

[5] R. Ramakrishnan *et al.*, "Quantum chemistry structures and properties of 134 kilo molecules," *Scientific Data*, vol. 1, p. 140022, 2014.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2017.

[7] V. P. Dwivedi *et al.*, "Benchmarking graph neural networks," *arXiv preprint arXiv:2003.00982*, 2020.

[8] P. Veličković *et al.*, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2018.

[9] D. Aharonov *et al.*, "Quantum walks on graphs," in *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001.

[10] A. Vlasic and S. Aguinaga, "Quop: A quantum operator representation for nodes," in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 1. IEEE, 2025, pp. 338–348.

[11] R. Sato *et al.*, "Qwalkvec: Node embedding by quantum walk," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2024, pp. 93–104.

[12] S. Thabet *et al.*, "Quantum positional encodings for graph neural networks," *arXiv preprint arXiv:2406.06547*, 2024.

[13] K. Xu *et al.*, "How powerful are graph neural networks?" in *International Conference on Learning Representations (ICLR)*, 2019.

[14] C. Ying *et al.*, "Do transformers really perform badly for graph representation?" *arXiv preprint arXiv:2106.05234*, 2021.

[15] V. Bergholm *et al.*, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," *arXiv preprint arXiv:1811.04968*, 2018.

[16] J. R. McClean *et al.*, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, no. 1, p. 4812, 2018.