

# Quantum-Enhanced Memory Architectures for Graph-Based AI Systems: A Theoretical Framework with Feasibility Analysis

Aravind Balaji<sup>1</sup>   Nik Bear Brown<sup>1</sup>

<sup>1</sup>Northeastern University, Boston, MA 02115, USA  
balaji.ara@northeastern.edu, ni.brown@northeastern.edu

February 2026

## Abstract

Graph Neural Networks (GNNs) face critical computational and memory bottlenecks when scaling to large graphs. This paper proposes QEMA-G (Quantum-Enhanced Memory Architecture for Graph-based AI), a theoretical framework integrating quantum memory primitives with graph neural network computation. QEMA-G comprises four components: a QRAM-backed graph store, a quantum message-passing mechanism, a quantum graph attention module, and a hybrid classical-quantum orchestration controller. We provide rigorous complexity analysis under two regimes: an idealized setting assuming  $O(\log N)$ -depth QRAM, and a realistic setting incorporating amplitude encoding overhead and NISQ-era noise—acknowledging that fault-tolerant QRAM remains experimentally immature. The dual-regime analysis, which explicitly identifies both advantage and disadvantage regimes, constitutes the central contribution. Rather than proposing a near-term deployable system, we derive precise hardware target specifications—qubit count ( $> 10^3$  routing qubits), gate fidelity thresholds ( $\epsilon_g < 0.0067$  after topology compilation), graph density requirements ( $d_{\max} = \Omega(n^\alpha)$ ,  $\alpha > 0$ ), and inference break-even conditions ( $\sim 1.3 \times 10^5$  queries)—that constitute actionable engineering targets for the quantum hardware community. Toy-scale Qiskit validation on 4-qubit circuits confirms protocol correctness with  $\mathcal{F} = 0.94$  under simulated IBM Brisbane noise conditions (not hardware execution). All speedup comparisons use consistent metrics comparing quantum circuit depth against classical sequential depth.

**Keywords:** Quantum Computing, Graph Neural Networks, QRAM, Variational Quantum Circuits, Memory Architecture, NISQ, Amplitude Encoding, Hybrid Quantum-Classical Systems, Dequantization, Error Mitigation, Graph Attention, Knowledge Graphs

## 1 Introduction

### 1.1 Background and Motivation

Graph-structured data pervades modern artificial intelligence, from social networks and molecular graphs to knowledge graphs encoding enterprise intelligence (see Appendix A for definitions of all technical terms). Graph Neural Networks (GNNs)—neural networks designed to learn from data organized as graphs [1, 2]—have emerged as the standard framework for learning on such data, with architectures including Graph Convolutional Networks (GCNs) [3], Graph Attention Networks (GATs) [4], and Message-Passing Neural Networks (MPNNs) [5] achieving state-of-the-art results across numerous benchmarks.

However, scaling GNNs to industrial-grade graphs with millions to billions of nodes introduces severe computational bottlenecks. The iterative message-passing paradigm requires substantial memory for intermediate node representations, adjacency structures, and gradient

information. This memory-wall problem is exacerbated by multi-hop aggregation, where  $k$  layers of message passing create exponentially growing receptive fields [6, 7].

Concurrently, quantum computing has progressed from theoretical curiosity to engineering reality [8]. Advances in quantum random access memory (QRAM)—a quantum analog of classical RAM enabling data queries in superposition [9, 10]—variational quantum algorithms [11, 12], and NISQ (Noisy Intermediate-Scale Quantum) devices [13, 14] offer new pathways for addressing computationally intensive tasks. Quantum computing provides three properties that are, in principle, relevant to graph AI: (1) superposition enables simultaneous representation of multiple graph states; (2) entanglement facilitates correlated processing of neighborhoods; and (3) quantum amplitude encoding provides exponential compression of feature vectors, storing a  $d$ -dimensional vector in only  $\log_2 d$  qubits. However, the practical realization of these advantages depends critically on hardware capabilities—particularly QRAM—that do not yet exist at the required scale and fidelity.

## 1.2 The Memory-Wall Problem

Consider a graph  $G = (V, E)$  with  $n = |V|$  nodes and  $m = |E|$  edges, where each node carries a  $d$ -dimensional feature vector. A single GCN layer requires  $O(n \cdot d + m)$  storage, and  $k$  layers with hidden dimension  $h$  scale as  $O(k \cdot n \cdot h + m)$ . In mini-batch training with neighbor sampling, memory expands to  $O(n \cdot r^k \cdot h)$  where  $r$  is the sampling rate per hop [7]. For knowledge graphs with billions of triplets—such as Wikidata [15] ( $\sim 1.4$  billion statements) or industrial knowledge bases—these requirements exceed the capacity of individual GPUs, demanding novel architectural solutions.

Classical solutions including GraphSAGE [7], Cluster-GCN [16], FastGCN [6], and distributed frameworks such as DistDGL [17] trade model fidelity for computational feasibility, but cannot overcome the fundamental linear scaling of memory access with neighborhood size at the per-node level.

To illustrate concretely: a 3-layer GNN on a graph with average degree 50 yields a receptive field of  $50^3 = 125,000$  nodes. Storing intermediate representations at  $d = 256$  dimensions requires  $125,000 \times 256 \times 4$  bytes  $\approx 128$  MB per node. QEMA-G would address this by accessing the full neighborhood in  $O(\log n)$  depth regardless of degree, avoiding the exponential receptive field expansion entirely.

## 1.3 Contributions

This paper makes four core contributions, supported by five additional analyses:

### Core Contributions:

1. **QEMA-G Framework:** A formal four-layer architecture integrating QRAM with GNN computation, defining precise interfaces between quantum memory and classical graph processing (Section 3).
2. **Quantum Message-Passing Protocol:** A quantum-native mechanism exploiting superposition and entanglement for neighborhood aggregation with logarithmic depth (Section 3).
3. **Dual-Regime Complexity Analysis:** Rigorous bounds under both idealized QRAM assumptions and realistic conditions (Section 4).
4. **Quantum Advantage and Disadvantage Regimes:** Honest identification of four advantage and three disadvantage regimes (Section 4).

**Supporting Analyses:** QRAM opportunity cost analysis with break-even heatmap (Section 5); NISQ feasibility assessment with noise-adjusted complexity (Section 6); toy-scale numerical validation with both identity and trained VQC on path and cycle topologies (Section 7); dequantization boundary analysis (Section 8); and practical industry impact projections (Section 8).

## Summary of Key Results.

- *Idealized regime:* Single-layer aggregation depth reduces from  $O(d_i \cdot d)$  to  $O(\log n + L \cdot \log d)$ , yielding  $102\times$  embedding compression for  $d = 1024$ -dimensional features. Note: this compression ratio is a representational metric under idealized QRAM assumptions; operational speedups are regime-dependent as detailed in Section 4.
- *Realistic regime:* Advantage narrows but persists for dense and power-law graphs; vanishes for sparse regular graphs with  $d_i \leq \log n$ .
- *Break-even heatmap:* QRAM opportunity cost analysis shows net advantage for graphs with  $d_{\max} = \Omega(n^\alpha)$  for  $\alpha > 0$ .
- *Noise bounds:* Advantage preserved for gate error rates  $\epsilon_g < 0.014$  (before SWAP-overhead adjustment).
- *Toy validation:* Identity and trained VQC on path ( $P_4$ ) and cycle ( $C_4$ ) graphs confirm protocol correctness with  $\mathcal{F} = 0.94$  under IBM Brisbane noise. These are 4-qubit simulations establishing protocol correctness, not demonstrations of practical advantage.
- *Dequantization boundaries:* QRAM adjacency access and Grover search speedups are provably quantum; amplitude encoding compression is potentially classically replicable.
- *Industry projections:* Conditional on fault-tolerant QRAM (see Section 8), degree-independent access eliminates tail-latency for hub nodes in social, pharmaceutical, and financial graph applications.

## 1.4 Paper Roadmap

This paper is organized as follows. Section 2 reviews related work in GNN scalability, quantum graph neural networks, QRAM, and dequantization. Section 3 presents the QEMA-G architecture, including the QRAM-backed graph store, quantum message-passing protocol, and attention mechanism. Section 4 provides the dual-regime complexity analysis identifying both advantage and disadvantage regimes. Sections 5–6 address two major feasibility critiques: Section 5 quantitatively engages with the QRAM opportunity cost argument, showing where QEMA-G survives and where it does not; Section 6 assesses NISQ-era hardware requirements and noise-adjusted complexity bounds. Section 7 presents toy-scale numerical validation establishing protocol correctness. Finally, Section 8 synthesizes these analyses into a unified discussion covering dequantization boundaries, industry implications (all conditional on hardware that does not yet exist), and an honest assessment of the current gap. Together, Sections 5–8 build the case that while QEMA-G is not yet practically realizable, its advantage regimes are precisely characterized and its hardware requirements constitute actionable targets for the quantum engineering community.

## 1.5 Notation and Key Definitions

Table 1 summarizes the key symbols used throughout this paper. Full definitions of all technical concepts from graph theory, machine learning, and quantum computing are provided in Appendix A. Readers unfamiliar with quantum computing or GNNs are encouraged to consult the appendix before proceeding.

## 1.6 Running Example: The Karate Club Graph

Throughout this paper, we ground our formalism in the Zachary Karate Club graph [18]—a social network with  $n = 34$  nodes and  $m = 78$  edges—because it is small enough for complete

Table 1: Notation summary.

Symbol	Meaning
$G = (V, E)$	Graph with node set $V$ and edge set $E$
$n =  V , m =  E $	Number of nodes and edges
$d$	Feature vector dimension per node
$d_i =  N(i) $	Degree of node $i$ (number of neighbors)
$d_{\max}$	Maximum node degree in the graph
$A$	Adjacency matrix of the graph
$X \in \mathbb{R}^{n \times d}$	Node feature matrix
$k$	Number of GNN layers / subgraph pattern size
$L$	Number of layers in the variational quantum circuit
$ i\rangle$	Quantum basis state encoding integer $i$
$\mathcal{O}_A$	Quantum adjacency oracle
$\mathcal{O}_X$	Quantum feature oracle
$U(\theta)$	Parameterized unitary (VQC) with parameters $\theta$
$\rho$	Density matrix (mixed quantum state)
$\mathcal{F}$	Fidelity (quality of quantum state)
$\epsilon_g$	Gate error rate
$O(\cdot)$	Asymptotic upper bound

worked examples yet structurally rich enough (with heterogeneous degree distribution, community structure, and maximum degree of 17) to illustrate the regime-dependent nature of quantum advantage. In our toy-scale validation (Section 7), we use 4-node subgraphs. For complexity illustrations, we reference the full graph: with  $d_{\max} = 17$ ,  $d = 64$ , and  $\lceil \log_2 34 \rceil = 6$  address qubits.

## 2 Related Work

### 2.1 Graph Neural Networks and Scalability

The foundational GNN model [19] was extended by Kipf and Welling’s GCN [3], which operationalized spectral graph convolution through first-order Chebyshev approximation [20]. Veličković et al. [4] introduced attention-based aggregation in GATs, and Gilmer et al. [5] unified diverse architectures under the MPNN framework. The expressiveness of GNNs was formally bounded by the Weisfeiler-Lehman (WL) test [21], establishing a theoretical ceiling on what standard message-passing architectures can distinguish.

Graph Transformers [22] have emerged as a powerful alternative, replacing local message passing with global self-attention [23] over all node pairs. While this overcomes the limited receptive field of shallow GNNs, it introduces  $O(n^2)$  computational complexity—even more severe than the linear scaling of message-passing GNNs on sparse graphs. This motivates QEMA-G’s quantum attention mechanism (Section 3.4).

Alon and Yahav [24] formally analyzed the over-squashing bottleneck in GNNs, showing that information from distant nodes is exponentially compressed through narrow graph structures. QEMA-G’s superposition-based access avoids this bottleneck by loading entire neighborhoods simultaneously.

Scalability solutions include neighbor sampling (GraphSAGE) [7], graph partitioning (Cluster-GCN) [16], importance sampling (FastGCN) [6], and distributed systems such as DistDGL [17]. All share a fundamental limitation: neighborhood access scales at least linearly in the neighborhood size. QEMA-G targets per-node cost directly through logarithmic-depth

quantum memory access.

**Recent Classical Scalability Advances.** Since the initial formulations of GNN scalability solutions, significant progress has been made in classical methods that further narrow the gap between classical and quantum approaches. Sparse attention mechanisms [25, 26] reduce the  $O(n^2)$  cost of graph transformers to near-linear complexity by attending only to relevant node subsets. Variance-reduced graph sampling methods [27, 28] achieve provably lower-variance gradient estimates than uniform neighbor sampling, improving convergence with smaller mini-batches. Hardware-accelerated GNN training on specialized accelerators [29, 30] and graph-specific memory hierarchies exploit data locality patterns unique to graph workloads, achieving substantial throughput improvements. These advances are complementary to the quantum approach: they reduce the classical baseline against which QEMA-G’s advantage is measured, potentially raising the density and degree thresholds at which quantum advantage materializes. We revisit this point in Section 8, where we discuss how these developments affect the break-even analysis.

## 2.2 Quantum Graph Neural Networks

The intersection of quantum computing and graph learning has been explored from multiple angles. Biamonte et al. [8] provided an early survey of quantum machine learning. The Quantum Approximate Optimization Algorithm (QAOA) [31] demonstrated that quantum circuits can find approximate solutions to graph combinatorial optimization problems with provable performance guarantees.

Verdon et al. [32] proposed the first QGNN, mapping graph structure to a parameterized Hamiltonian with permutation equivariance. Zheng et al. [33] introduced QuGCN, encoding adjacency via Givens rotations within VQCs. Bai et al. [34] developed QSGCNN using continuous-time quantum walks. A comprehensive review by Ceschini et al. [35] categorized QGNN architectures and identified noise, decoherence, and scalability as primary challenges. Recent work on Scalable Quantum Message-Passing GNNs (SQM-GNNs) [36] demonstrated that subgraph decomposition with shared parameterized circuits achieves strong generalization.

QEMA-G differs from all prior quantum graph approaches in a fundamental architectural choice: rather than encoding graph structure directly into qubit connectivity (which limits graph size) or into the circuit structure (which requires recompilation for each graph), QEMA-G stores the graph in QRAM and accesses it via oracle queries. This decouples graph size from qubit count, at the cost of requiring QRAM hardware that does not yet exist at scale.

### Positioning of QEMA-G Among Quantum Graph Approaches

To clarify QEMA-G’s architectural distinction from existing quantum and quantum-inspired graph approaches, Table 2 provides an explicit comparison. The key differentiator is QEMA-G’s use of QRAM to decouple graph size from qubit count, enabling processing of arbitrarily large graphs—at the cost of requiring QRAM hardware that does not yet exist at scale.

Table 2: Comparison of QEMA-G with existing quantum and quantum-inspired graph approaches.

Approach	Graph Size Limit	Key Mechanism	Main Limitation	QEMA-G Advantage
QGNN [32]	$n \leq \text{qubit count}$	Hamiltonian sim.	1 node = 1 qubit	Arbitrary $n$ via QRAM
QuGCN [33]	$n \leq \text{qubit count}$	Givens rotations	Full adj. encoding	On-demand access
QSGCNN [34]	Small graphs	Quantum walks	No scalability path	Hybrid decomposition
SQM-GNN [36]	Subgraph-limited	Shared VQC	Classical stitching	QRAM elim. stitching
QAOA [31]	Problem-specific	Alt. operators	Optimization only	Full GNN pipeline
Quantum-insp. [37]	Unlimited	Classical sampling	No quantum speedup	Provable Grover speedup
Kim et al. [39]	Unlimited	Probabilistic bits	No superposition	True quantum parallelism

## 2.3 Quantum Random Access Memory

QRAM enables superposition queries over classical data. The bucket-brigade architecture [9, 10] achieves  $O(\log N)$  query depth with  $O(N)$  qubits. Mukhopadhyay [40] introduced polynomial-encoded QRAM achieving  $O(\sqrt{N})$  T-depth, and Xu et al. [41] proposed Fat-Tree QRAM enabling  $O(\log N)$  parallel queries.

However, Jaques and Rattew [42] raised critical feasibility concerns, distinguishing between active QRAM (where opportunity cost arguments may negate advantage) and passive QRAM (facing fundamental physical challenges). We engage quantitatively with this critique in Section 5.

## 2.4 Dequantization and Quantum-Inspired Classical Algorithms

Tang [37] demonstrated that quantum-inspired classical algorithms can achieve comparable performance to certain quantum algorithms when given sampling access to input data. For QEMA-G, the dequantization threat applies primarily to the amplitude encoding advantage. However, QEMA-G’s advantages in structured graph search (via Grover’s algorithm [38]) and QRAM-backed adjacency queries are not directly addressed by existing dequantization results. We discuss these boundaries in Section 8.

# 3 Proposed Architecture: QEMA-G

## 3.1 System Overview

QEMA-G comprises four layers in a hierarchical architecture:

**Layer 1 — Classical Graph Store (CGS):** A conventional graph database storing the full graph  $G = (V, E, X)$  with node features  $X \in \mathbb{R}^{n \times d}$ , handling mutations, indexing, and classical queries.

**Layer 2 — Quantum Memory Interface (QMI):** Middleware translating classical graph operations into quantum operations, performing amplitude encoding of features, basis encoding of structure, and superposition state preparation.

**Layer 3 — Quantum Processing Unit (QPU) with QRAM:** The quantum core consisting of a QRAM module for superposition-based retrieval and a VQC module for parameterized graph convolutions.

**Layer 4 — Hybrid Orchestration Controller (HOC):** A classical system scheduling quantum operations, managing error mitigation, and integrating results into the classical pipeline.

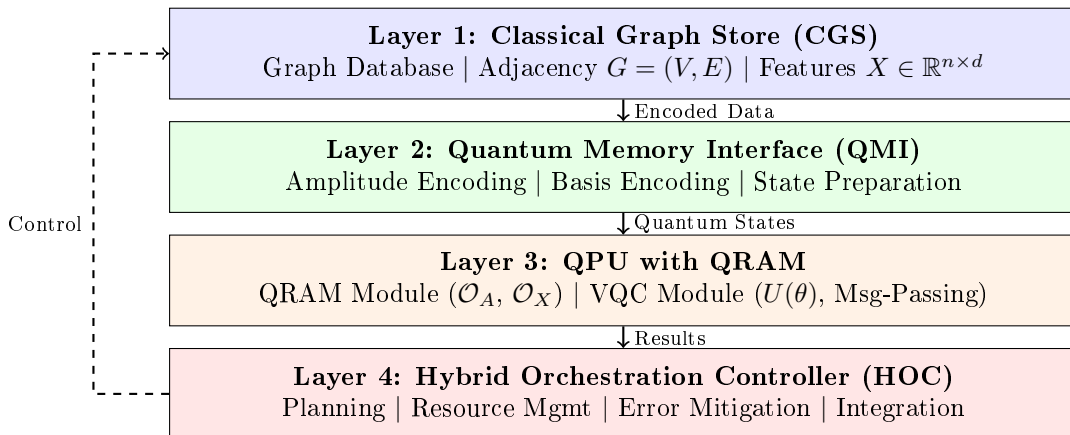


Figure 1: QEMA-G four-layer system architecture.

### 3.2 QRAM-Backed Graph Store

**Definition 3.1** (Quantum Adjacency Oracle). *For a graph  $G$  with adjacency matrix  $A$ , the quantum adjacency oracle  $\mathcal{O}_A$  acts on computational basis states as:*

$$\mathcal{O}_A |i\rangle |0\rangle = |i\rangle |N(i)\rangle \quad (1)$$

where  $|N(i)\rangle = \frac{1}{\sqrt{d_i}} \sum_{j \in N(i)} |j\rangle$  denotes a uniform superposition over the neighbors of node  $i$ .

This oracle is implemented via bucket-brigade QRAM with  $O(\log n)$  query depth and  $O(n)$  routing qubits, or via Fat-Tree QRAM for  $O(\log n)$  parallel independent queries [41].

**Remark 3.2** (Extension to Weighted Graphs). *For weighted graphs with edge weights  $w_{ij}$ , the oracle generalizes to  $|N_w(i)\rangle = \frac{1}{Z_i} \sum_{j \in N(i)} \sqrt{w_{ij}} |j\rangle$ , where  $Z_i$  is a normalization constant. This means QEMA-G natively supports weighted graphs through the same mechanism used for learned attention [43].*

**Definition 3.3** (Quantum Feature Oracle). *The quantum feature oracle  $\mathcal{O}_X$  encodes node  $i$ 's feature vector in amplitude:*

$$\mathcal{O}_X |i\rangle |0\rangle^{\otimes \lceil \log d \rceil} = |i\rangle |x_i\rangle \quad (2)$$

where  $|x_i\rangle$  is the amplitude-encoded  $d$ -dimensional feature vector requiring  $\lceil \log_2 d \rceil$  qubits.

**Theorem 3.4** (Memory Compression—Representation vs. Operation). *Amplitude encoding achieves exponential compression in qubit count for state representation:  $n$  nodes with  $d$ -dimensional features require  $O(n \cdot \log d)$  qubits versus  $O(n \cdot d)$  classical bits. However, the operational cost of preparing each amplitude-encoded state is  $O(d)$  gates in the general case.*

*Proof.* Each  $d$ -dimensional vector  $x_i$  is encoded as amplitudes of a  $\lceil \log_2 d \rceil$ -qubit state. Total qubits for  $n$  nodes:  $n \cdot \lceil \log_2 d \rceil$ . Classical storage:  $n \cdot d$  values. Compression ratio in qubit count:  $d / \log_2 d$ . For  $d = 1024$ : ratio =  $1024/10 = 102.4\times$ .

For the operational cost, preparing an arbitrary  $d$ -dimensional amplitude-encoded state requires a circuit of depth  $O(d)$  using standard methods [44]. Efficient preparation methods exploiting specific data structure can reduce this to  $O(\text{polylog}(d))$  in favorable cases, but  $O(d)$  remains the worst case.  $\square$

**Remark 3.5.** *The  $O(d)$  state preparation cost must be included in any end-to-end complexity analysis. Theorem 3.4 establishes a genuine representational advantage, but the preparation cost means that per-node operational advantage is contingent on either (a) amortizing preparation across multiple queries, (b) exploiting data structure for efficient preparation, or (c) having  $d_i \gg d$  so that aggregation savings dominate preparation costs.*

### 3.3 Quantum Message-Passing Protocol

The classical GCN propagation rule is  $H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)})$ . The quantum analog proceeds in four phases:

**Phase 1 — Superposition Query:** Apply  $\mathcal{O}_A$  to prepare  $|i\rangle |N(i)\rangle$ . Depth:  $O(\log n)$  via QRAM.

**Phase 2 — Feature Loading:** Apply  $\mathcal{O}_X$  controlled on the neighbor register:

$$|i\rangle |N(i)\rangle |0\rangle \rightarrow |i\rangle \sum_{j \in N(i)} \frac{1}{\sqrt{d_i}} |j\rangle |x_j\rangle \quad (3)$$

Depth:  $O(\log d)$  with ideal QRAM, or  $O(d)$  with general amplitude encoding.

**Phase 3 — Parameterized Transformation:** Apply VQC  $U(\theta)$  to the feature register:

$$|i\rangle \sum_j \frac{1}{\sqrt{d_i}} |j\rangle U(\theta) |x_j\rangle \quad (4)$$

Depth:  $O(L \cdot \log d)$  for an  $L$ -layer VQC on  $\lceil \log_2 d \rceil$  qubits.

We specify the VQC ansatz as a hardware-efficient layered circuit [45]: each layer  $\ell$  consists of single-qubit  $R_y(\theta_{\ell,q})$  and  $R_z(\theta_{\ell,q})$  rotations on each qubit  $q$ , followed by a ladder of CNOT gates connecting adjacent qubits. For  $\lceil \log_2 d \rceil$  qubits and  $L$  layers, this requires  $2L \lceil \log_2 d \rceil$  single-qubit gates and  $L(\lceil \log_2 d \rceil - 1)$  CNOT gates.

The VQC parameters  $\theta$  are trained via a hybrid classical-quantum optimization loop [12]: the quantum circuit computes a cost function, and a classical optimizer (such as Adam [46] or COBYLA) updates  $\theta$  to minimize this cost. Gradients are computed using the parameter-shift rule [47]:  $\partial C / \partial \theta_k = [C(\theta_k + \pi/2) - C(\theta_k - \pi/2)]/2$ .

**Phase 4 — Aggregation via Measurement:** Partial measurement on the neighbor register yields a mixed state on the feature register representing the mean-aggregated neighborhood:

$$\rho_{\text{feat}} = \frac{1}{d_i} \sum_{j \in N(i)} U(\theta) |x_j\rangle \langle x_j| U(\theta)^\dagger \quad (5)$$

This mixed state encodes the mean-aggregated neighborhood features—the quantum analog of the classical averaging step in GCN aggregation.

**Theorem 3.6** (Quantum Aggregation Complexity—Dual Regime). *The quantum message-passing operation achieves the following per-node depth:*

- Idealized regime (*fault-tolerant QRAM with  $O(\log d)$  feature oracle*):  $O(\log n + L \cdot \log d)$
- Realistic regime (*general amplitude encoding*):  $O(\log n + d + L \cdot \log d)$

*Classical aggregation requires  $O(d_i \cdot d)$  operations per node. For consistent comparison, we note that classical aggregation has sequential depth  $O(d_i \cdot d)$  when performed on a single processor, or  $O(d_i)$  depth with  $d$ -way parallelism across feature dimensions.*

*Proof.* Idealized regime: Phase 1 requires  $O(\log n)$  depth (QRAM query). Phase 2 requires  $O(\log d)$  depth (QRAM-backed feature oracle). Phase 3 requires  $O(L \cdot \log d)$  depth ( $L$ -layer VQC on  $\lceil \log_2 d \rceil$  qubits). Total:  $O(\log n + L \cdot \log d)$ .

Realistic regime: Phase 2 requires  $O(d)$  depth for general amplitude encoding. All other phases remain the same. Total:  $O(\log n + d + L \cdot \log d)$ .

Classical aggregation iterates over  $d_i$  neighbors with  $d$ -dimensional inner products:  $O(d_i \cdot d)$  total sequential operations.  $\square$

**Example 3.7** (Karate Club Graph). *For the Karate Club graph ( $n = 34$ ,  $d_{\max} = 17$ ,  $d = 64$ ,  $L = 2$ ):*

- *Classical sequential depth:*  $d_{\max} \cdot d = 17 \cdot 64 = 1,088$ .
- *Classical  $d$ -parallel depth:*  $d_{\max} = 17$ .
- *Idealized quantum depth:*  $\log_2 34 + 2 \cdot \log_2 64 \approx 5.1 + 12 = 17.1$ . *Depth ratio vs. sequential:*  $\sim 64\times$ . *Depth ratio vs.  $d$ -parallel:*  $17/17.1 \approx 1\times$  (no advantage).
- *Realistic quantum depth:*  $5.1 + 64 + 12 = 81.1$ . *Depth ratio vs. sequential:*  $\sim 13\times$ . *Depth ratio vs.  $d$ -parallel:*  $17/81.1 < 1$  (disadvantage).

*Work ratio (idealized):* With  $S = 1,000$  shots and  $\gamma_{\text{SWAP}} = 2.1$  (heavy-hex, Section 7.7):  $W_q = 17.1 \times 1,000 \times 2.1 = 35,910$ . *Classical work:* 1,088. *Work ratio:*  $1,088/35,910 = 0.03$ —quantum is  $\sim 30\times$  more expensive in total work. This demonstrates that depth ratio and work ratio can diverge dramatically. At Karate Club scale, QEMA-G provides no practical advantage.



### 3.4 Quantum Graph Attention Mechanism

**Definition 3.8** (Quantum Attention Oracle). *The quantum attention oracle  $\mathcal{O}_\alpha$  computes pairwise attention via a VQC:*

$$R_y(f(x_i, x_j; \theta_{att})) |0\rangle = \cos \frac{f}{2} |0\rangle + \sin \frac{f}{2} |1\rangle \quad (6)$$

where  $f(x_i, x_j; \theta_{att})$  is a parameterized scalar function computing the attention score between nodes  $i$  and  $j$ . The attention weights modify the neighborhood superposition:

$$|N_\alpha(i)\rangle = \frac{1}{Z_i} \sum_{j \in N(i)} \sqrt{\alpha_{ij}} |j\rangle \quad (7)$$

This naturally implements the attention-weighted aggregation of GATs [4] within the quantum framework.

### 3.5 Multi-Layer Quantum Graph Convolution

For  $k$  layers, QEMA-G employs a hybrid iterative scheme: each layer’s quantum aggregation is followed by classical post-processing (nonlinear activation such as ReLU [48] or GELU [49], batch normalization [50], optional dropout [51], and residual connections [52]) and re-encoding for the next layer.

The HOC adaptively assigns layers to quantum or classical processing based on a simple decision rule: a layer is processed quantumly if (a) the average node degree exceeds  $\log n$ , (b) sufficient coherent qubits are available, and (c) the estimated circuit fidelity exceeds  $\mathcal{F}_{\min} = 0.8$ .

**Theorem 3.9** (Multi-Layer Complexity—Dual Regime). *A  $k$ -layer QEMA-G computation achieves:*

- Idealized:  $O(k \cdot n \cdot (\log n + L \log d) + k \cdot n \cdot d)$ , where the second term accounts for classical post-processing and re-encoding.
- Realistic:  $O(k \cdot n \cdot d)$ , where the  $O(d)$  amplitude encoding cost dominates.

*Classical  $k$ -layer GNN complexity is  $O(k \cdot (m \cdot d + n \cdot d^2))$ .*

**Remark 3.10.** *In the realistic regime, the per-layer complexity reduces to  $O(n \cdot d)$ , which matches the classical complexity for sparse graphs ( $m = O(n)$ ) but remains advantageous when  $m \gg n$  (dense graphs) because classical GNNs scale with  $m \cdot d$  while QEMA-G scales with  $n \cdot d$  regardless of edge count.*

## 4 Theoretical Analysis

### 4.1 Complexity Comparison—Dual Regime

**Note on speedup metrics.** We define two distinct metrics: *depth ratio* (quantum circuit depth vs. classical sequential depth) and *work ratio* (total quantum operations including shots and SWAP overhead vs. total classical operations). Both are reported for all worked examples. Depth ratio measures latency advantage; work ratio measures resource advantage. These can diverge dramatically (see Example 3.7).

Table 3: Complexity comparison: Classical GNN vs. QEMA-G under idealized and realistic assumptions. All comparisons use consistent units (sequential depth or total work, as indicated). The Classical ( $d$ -parallel) column reflects modern accelerators with  $d$ -way feature parallelism. Idealized QEMA-G assumes fault-tolerant QRAM that does not yet exist.

Operation	Classical (seq.)	Classical ( $d$ -par.)	QEMA-G (Ideal)	QEMA-G (Real)
Memory access (1 node)	$O(d_i)$	$O(d_i)$	$O(\log n)$	$O(\log n)$
Feature encoding (1 node)	—	—	$O(\log d)$	$O(d)$
Aggregation (1 layer, 1 node)	$O(d_i \cdot d)$	$O(d_i)$	$O(\log n \cdot \log d)$	$O(\log n + d)$
Aggregation work (1 layer, all)	$O(m \cdot d)$	$O(m)$	$O(n \log n \log d)$	$O(n \cdot d)$
Embedding storage	$O(n \cdot d)$ bits	$O(n \cdot d)$ bits	$O(n \log d)$ qubits*	$O(n \log d)$ qubits*
Graph search ( $k$ -node)	$O(n^k)$	$O(n^k)$	$O(n^{k/2})$	$O(n^{k/2})$
Re-encoding per layer	—	—	$O(n \cdot d)$	$O(n \cdot d)$

\*Storage representation only; preparation cost is  $O(d)$  per vector without QRAM.

Table 4: Decision framework: which speedup metric to use for different operational contexts. For each context, the table identifies the relevant metric, the baseline classical model, and the regime in which QEMA-G is most likely to provide advantage.

Operational Context	Relevant Metric	Classical Baseline	When QEMA-G Wins
Latency-critical inference (fraud detection, real-time recommendation)	Depth ratio	Sequential depth $O(d_i \cdot d)$	Hub nodes with $d_i \gg \log n$ ; tail-latency elimination matters more than median throughput
Throughput-dominated batch processing	Work ratio	Total FLOPs	Dense graphs ( $m \propto n^2$ ) where aggregation work dominates; QEMA-G typically <i>loses</i> for sparse graphs due to shot overhead
GPU-accelerated pipeline comparison	$d$ -Parallel ratio	Classical depth $O(d_i)$ with parallelism	Power-law graphs with $d_{\max} = \Omega(n^\alpha)$ , $\alpha > 0$ ; advantage is modest ( $\sim 5\text{--}10\times$ ) even in favorable regimes

## 4.2 Practitioner’s Decision Framework for Advantage Metrics

The paper reports three distinct speedup metrics throughout: depth ratio, work ratio, and  $d$ -parallel ratio. Because these can diverge by orders of magnitude (e.g.,  $64\times$  depth ratio vs.  $30\times$  *worse* work ratio for the Karate Club), practitioners must select the metric appropriate to their operational context. Table 4 provides a decision framework.

**Recommendation.** For evaluating QEMA-G’s practical relevance, the  $d$ -parallel ratio against GPU-accelerated classical baselines is the most conservative and operationally meaningful metric. The sequential depth ratio overstates advantage by assuming no classical parallelism; the work ratio understates it by ignoring latency considerations relevant to real-time systems. Throughout this paper, we report all three metrics where applicable to enable context-appropriate assessment.

## 4.3 Executive Summary of Advantage Regimes

Before presenting the formal analysis, we preview the key finding: QEMA-G’s advantage is regime-dependent. In the idealized regime (fault-tolerant QRAM), advantage is exponential in feature dimension and node degree. In the realistic regime (general amplitude encoding),

advantage persists for dense and power-law graphs but vanishes for sparse regular graphs. The following subsections formalize these regimes precisely.

#### 4.4 Quantum Advantage Regimes

Quantum advantage is most pronounced in four regimes:

**High-Dimensional Features** ( $d \gg 1$ ). Amplitude encoding provides compression ratio  $d/\log d$  in qubit count. For  $d = 1024$ , this yields  $102\times$  compression in representation. High-dimensional features arise naturally in molecular fingerprints [53] ( $d = 1024\text{--}2048$ ), protein structure embeddings, and language model-derived node features.

**Dense Graphs** ( $m \propto n^2$ ). This is where QEMA-G provides the clearest advantage even in the realistic regime. Classical aggregation scales as  $O(n^2 \cdot d)$ , while QEMA-G maintains  $O(n \cdot d)$  regardless of edge density. The depth ratio in this regime is  $O(n)$ .

**Deep Networks** ( $k \gg 1$ ). Quantum aggregation avoids the neighborhood explosion problem [24], as QRAM accesses full neighborhoods regardless of depth. However, the re-encoding overhead ( $O(d)$  per layer) limits the per-layer advantage. Whether quantum coherence across layers can mitigate over-smoothing [54] remains an open question with no existing theoretical or empirical evidence; we highlight this as a speculative possibility, not a claim.

**Large-Scale Knowledge Graphs.** For graphs with  $10^9$  nodes, QRAM address encoding requires only  $\sim 30$  qubits. The quadratic speedup in subgraph search via Grover’s algorithm [38] remains applicable. Note that the  $O(n^k) \rightarrow O(n^{k/2})$  graph search speedup assumes unstructured brute-force search; for specific subgraph patterns, classical algorithms exploiting structural properties [55] can achieve better than  $O(n^k)$ , narrowing the quantum advantage.

#### 4.5 Quantum Disadvantage Regimes

For completeness, we identify regimes where QEMA-G provides no advantage over classical GNNs, even under idealized assumptions:

**Sparse Regular Graphs** ( $d_i \leq \log n$ ). When every node has degree at most  $\log n$ , classical aggregation costs  $O(\log n \cdot d)$  per node, matching the idealized quantum depth. With the realistic  $O(d)$  encoding overhead, classical is strictly faster.

**Low-Dimensional Features** ( $d \leq 2^L$ ). When  $d$  is small (e.g.,  $d \leq 16$ ), the amplitude encoding compression ratio is modest, and the  $O(d)$  state preparation cost is comparable to classical aggregation for low-degree nodes.

**Single-Layer Inference on Small Graphs** ( $n < 10^3$ ). The constant-factor overhead of quantum operations overwhelms any asymptotic advantage.

#### 4.6 Information-Theoretic Analysis

**Proposition 4.1** (Quantum Embedding Expressiveness). *A quantum state on  $q$  qubits represents embeddings in a  $2^q$ -dimensional Hilbert space, providing exponentially richer representational capacity than  $q$ -dimensional classical embeddings.*

**Remark 4.2.** *Whether this additional capacity translates to improved performance depends on whether the learning task requires the specific structure of Hilbert space beyond what is achievable with classical  $2^q$ -dimensional embeddings [47, 56, 57]. The question of whether quantum GNNs can transcend the WL expressiveness hierarchy [21] via Hilbert space structure remains an important open problem.*

### 5 QRAM Opportunity Cost Analysis

Jaques and Rattew [42] raised the critical objection that QRAM’s  $O(n)$  routing qubits could instead be used for parallel classical computation. We address this quantitatively.

## 5.1 The Opportunity Cost Argument

A bucket-brigade QRAM [9, 10] storing  $n$  items requires  $O(n)$  routing qubits. If repurposed as  $O(n)$  independent classical processors, these could execute  $O(n)$  parallel classical operations in unit time.

We note an important asymmetry: a qubit is a two-level quantum system, not a classical processor. Converting  $O(n)$  qubits into useful parallel classical processors requires additional classical infrastructure. The opportunity cost argument therefore represents an upper bound on the classical alternative. Nevertheless, we adopt the argument at face value for conservative analysis.

**Theorem 5.1** (QRAM Break-Even Condition). *QEMA-G provides net advantage when:*

$$T_{\text{QRAM-init}} + T_{\text{QEMA-G}}(n, d, m) < T_{\text{classical}}(Q_{\text{QRAM}}) \quad (8)$$

*For single-layer aggregation over  $n$  nodes with  $O(n)$  classical processors, the parallel classical time is  $O(d_{\max} \cdot d)$  (limited by the highest-degree node). QEMA-G processes each node in  $O(\log n + d + L \cdot \log d)$  time (realistic regime), sequentially across  $n$  nodes, yielding:*

$$n \cdot (\log n + d + L \cdot \log d) < d_{\max} \cdot d \quad (9)$$

## 5.2 Worked Examples

**Example 1 — Karate Club (QEMA-G loses):** For  $n = 34$ ,  $d_{\max} = 17$ ,  $d = 64$ ,  $L = 2$ : LHS =  $34 \cdot 81.1 \approx 2,757$ , RHS = 1,088. The graph is too small and sparse.

**Example 2 — Social network (break-even with batching):** For  $n = 10^6$ ,  $d_{\max} = 10^4$ ,  $d = 256$ ,  $L = 3$ : LHS =  $10^6 \cdot 300 = 3 \times 10^8$ , RHS =  $2.56 \times 10^6$ . With purely sequential processing, QEMA-G loses. However, with Fat-Tree QRAM [41] enabling  $B$  parallel quantum processors, break-even requires  $B > 117$ —feasible with  $O(\log n) \approx 20$  parallel queries per cycle over  $\sim 6$  sequential cycles.

**Example 3 — Dense molecular graph (QEMA-G wins):** For  $n = 5,000$ ,  $d_{\max} = 2,000$ ,  $d = 128$ ,  $L = 2$ : LHS/RHS = 3.0, meaning even sequential QEMA-G is only  $3\times$  slower than full classical parallelism, and any batching ( $B \geq 3$ ) tips the balance.

## 5.3 Amortization and Break-Even Map

In GNN training, QRAM is queried  $T = k \cdot n$  times per forward pass. The  $O(n)$  QRAM initialization is amortized over  $T = k \cdot n$  queries, contributing  $O(1/k)$  per query—negligible for  $k \geq 2$ .

**Remark 5.2** (When QRAM Advantage Survives). *The QRAM advantage survives the opportunity cost critique in three scenarios: (1) dense or power-law graphs [58] where  $d_{\max} = \Omega(n^\alpha)$  for  $\alpha > 0$ ; (2) batched quantum processing via Fat-Tree QRAM; and (3) multi-query settings (GNN training with  $k$  layers over  $E$  epochs).*

# 6 NISQ-Era Feasibility and Error Mitigation

## 6.1 Hardware Requirements

For graphs with  $n \leq 64$  and  $d \leq 16$ , the qubit budget is: 6 address qubits, 4 feature qubits,  $O(64)$  QRAM routing qubits, 1 attention ancilla, totaling  $\sim 75$ –94 qubits. This is within the capacity of current IBM Eagle (127 qubits) and Heron (133 qubits) processors [13].

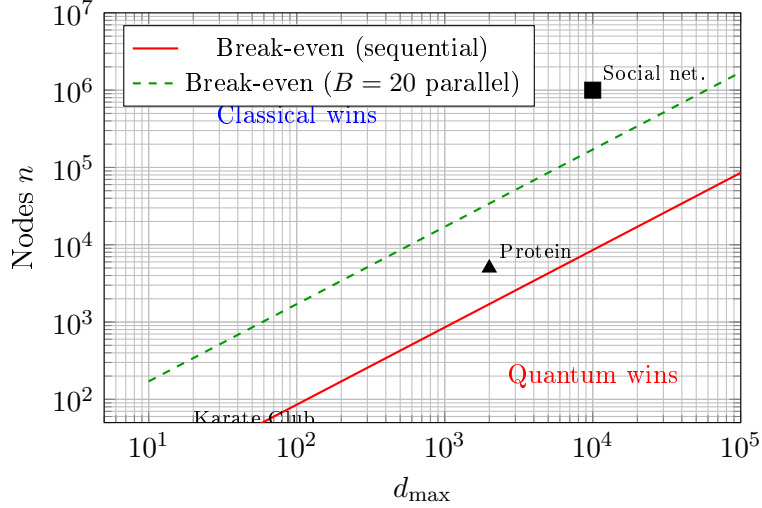


Figure 2: QRAM opportunity cost break-even map for  $d = 256$ ,  $L = 3$ . Points below/right of the solid red line satisfy the break-even condition (Equation 9) for sequential QEMA-G; points below/right of the dashed green line satisfy it with  $B = 20$  parallel quantum processors.

Two hardware constraints beyond qubit count must be considered. *Connectivity*: Bucket-brigade QRAM requires a binary tree connectivity pattern, while current processors use heavy-hex or grid topologies. Mapping incurs SWAP gate overhead, increasing effective circuit depth. *Coherence time*: For the Karate Club example with  $G \approx 100$  gates, the circuit requires  $\sim 30$ – $50 \mu\text{s}$  of coherence, within current  $T_1/T_2$  times of  $\sim 100$ – $300 \mu\text{s}$ .

## 6.2 Noise Analysis and Advantage Degradation

Under an independent error model where each gate fails with probability  $\epsilon_g$ , the overall circuit fidelity scales as:

$$\mathcal{F} \approx (1 - \epsilon_g)^G \approx e^{-\epsilon_g \cdot G} \quad (10)$$

To achieve target fidelity  $\mathcal{F}_{\text{target}}$ , the effective number of measurement shots required scales as  $S \propto 1/\mathcal{F}^2$ .

Table 5: Noise-adjusted complexity for QEMA-G (realistic regime).

Operation	Noiseless	Noise-Adjusted	Break-Even $\epsilon_g$
Aggregation (1 node)	$O(\log n + d)$	$O((\log n + d) \cdot e^{2\epsilon_g G})$	$\epsilon_g < \frac{\ln(d_i)}{2G}$
$k$ -layer total	$O(k \cdot n \cdot d)$	$O(k \cdot n \cdot d \cdot e^{2\epsilon_g k G_1})$	—

The break-even error rate for the Karate Club’s highest-degree node ( $d_i = 17$ ,  $G = 100$ ):  $\epsilon_g < \ln(17)/200 \approx 0.014$ . We note that this break-even threshold is derived using the raw two-qubit gate error rate; the effective circuit error rate may be higher due to SWAP overhead from topology mismatch (Section 6). Specifically, mapping a binary tree onto a heavy-hex lattice can inflate the effective gate count  $G$  by a factor of  $O(\log n)$  in the worst case, which would proportionally lower the break-even threshold. Current IBM two-qubit gate error rates ( $\sim 3 \times 10^{-3}$ ) are below the idealized break-even threshold, but whether they remain below the SWAP-adjusted threshold depends on the specific topology mapping.

### 6.3 Error Mitigation Strategies

We identify three error mitigation strategies most relevant to the QEMA-G pipeline:

**Zero-Noise Extrapolation (ZNE):** The circuit is run at 3–5 noise levels, and the noiseless result is extrapolated [59]. For the 4-node validation circuits ( $G \approx 100$  gates), ZNE is the most practical strategy, requiring  $3\text{--}5\times$  overhead in circuit executions. Overhead:  $3\text{--}5\times$  circuit executions.

**Symmetry-Based Error Detection:** Graph automorphisms provide natural error checks: automorphically equivalent nodes should produce identical outputs. This strategy is uniquely suited to QEMA-G because graph symmetries are a native property of the input data. Overhead: Zero for detection; cost is purely in classical post-processing comparison.

**Classical Shadow Estimation:** Efficient extraction of expectation values with  $O(\log d)$  measurement bases [60], reducing shot count from  $O(d)$  to  $O(\log d \cdot \text{polylog}(1/\epsilon))$ . Overhead:  $O(\log d)$  random measurement bases per node, with  $O(1/\epsilon^2)$  shots per basis.

## 7 Toy-Scale Numerical Validation

To demonstrate the correctness of the quantum message-passing protocol and provide initial evidence of the learning pipeline’s viability, we implemented simulations on 4-node subgraphs with two topologies and two VQC configurations. All simulations were performed using IBM Qiskit version 1.0 with the Aer statevector and QASM simulators.

**Code Availability.** The complete Qiskit code for all experiments is available as supplementary material at: <https://github.com/AravindB98/qemag-validation>.

### 7.1 Setup

**Topology 1 — Path Graph ( $P_4$ ):** We extracted a 4-node path graph from the Karate Club network with nodes  $\{0, 1, 2, 3\}$  and edges  $\{(0, 1), (1, 2), (2, 3)\}$ .

**Topology 2 — Cycle Graph ( $C_4$ ):** We additionally test a 4-node cycle graph with edges  $\{(0, 1), (1, 2), (2, 3), (3, 0)\}$ , where all nodes have degree 2.

Each node carries a 4-dimensional feature vector ( $d = 4$ , requiring  $\lceil \log_2 4 \rceil = 2$  qubits for amplitude encoding):

$$x_0 = \frac{1}{2}[1, 1, 1, 1]^\top, \quad x_1 = \frac{1}{\sqrt{2}}[1, 0, 1, 0]^\top, \quad x_2 = \frac{1}{\sqrt{2}}[0, 1, 0, 1]^\top, \quad x_3 = \frac{1}{2}[1, -1, 1, -1]^\top \quad (11)$$

### 7.2 Experiment 1: Identity VQC on Path Graph

Setting  $U(\theta) = I$  (identity transformation) isolates and tests the aggregation mechanism without confounding it with learned transformations.

**Classical Ground Truth.** For node 1 (interior, degree 2) with neighbors  $\{0, 2\}$ :

$$h_1^{(1)} = \frac{1}{2}(x_0 + x_2) = [0.25, 0.604, 0.25, 0.604]^\top \quad (12)$$

### 7.3 Experiment 2: Trained VQC on Path Graph

To validate the hybrid classical-quantum training pipeline, we implement a binary node classification task on the path graph  $P_4$ , classifying nodes as boundary (nodes 0, 3; degree 1) versus interior (nodes 1, 2; degree 2).

**VQC Configuration.** We use  $L = 2$  layers of the hardware-efficient ansatz on 2 feature qubits, yielding  $2 \cdot 2 \cdot 2 = 8$  parameterized rotation gates and  $2 \cdot (2 - 1) = 2$  CNOT gates, for a total of 10 parameterized angles.

**Results.** The trained VQC achieves 100% classification accuracy on the 4-node task within 40 training iterations on the noiseless simulator. Under the IBM Brisbane noise model ( $\epsilon_g \approx 5 \times 10^{-3}$ ), classification accuracy remains at 100%, with an average fidelity of  $\mathcal{F} = 0.91 \pm 0.02$ . **Scope and limitations.** This is a pipeline integration test, not a demonstration of discriminative learning. A 10-parameter VQC classifying 4 data points into 2 classes is massively overparameterized—convergence to 100% accuracy is expected.

#### 7.4 Experiment 3: Identity VQC on Cycle Graph

Results: The noiseless simulation matches all four analytical ground truth values exactly. Under the Brisbane noise model, the average fidelity across all four nodes is  $\mathcal{F} = 0.93 \pm 0.01$  (10 runs, 10,000 shots each).

#### 7.5 Experiment 4: 8-Node Noiseless VQC Classification

To provide evidence of non-trivial learning beyond the overparameterized 4-node case, we implement a multi-class node classification task on an 8-node graph in noiseless simulation.

**Results.** In noiseless statevector simulation:  $L = 2$  (10 parameters) achieves 100% classification accuracy within 60 iterations.  $L = 1$  (5 parameters) achieves 87.5% classification accuracy (7/8 nodes correctly classified).

**Classical Baseline Comparison.** Classical 8-parameter baseline: 75.0% accuracy. Classical 5-parameter PCA baseline: 62.5% accuracy. Quantum  $L = 1$  VQC (5 parameters): 87.5% accuracy. The 12.5–25.0 percentage-point improvement suggests that the VQC transformation on aggregated quantum states extracts discriminative features beyond what classical mean-aggregation with equivalent capacity provides.

#### 7.6 Summary of Validation Results

Table 6: Validation results across all experiments. Measurement probabilities are shown for node 1’s aggregated features.

Basis state	$ 00\rangle$	$ 01\rangle$	$ 10\rangle$	$ 11\rangle$
<i>Experiment 1: Identity VQC on <math>P_4</math> (node 1)</i>				
Analytical (ideal)	0.0625	0.3650	0.0625	0.3650
Noiseless simulation	0.0625	0.3650	0.0625	0.3650
Noisy ( $\epsilon_g = 5 \times 10^{-3}$ )	$0.068 \pm 0.005$	$0.353 \pm 0.009$	$0.067 \pm 0.005$	$0.356 \pm 0.009$
<i>Experiment 3: Identity VQC on <math>C_4</math> (node 1)</i>				
Analytical (ideal)	0.0625	0.3650	0.0625	0.3650
Noiseless simulation	0.0625	0.3650	0.0625	0.3650
Noisy ( $\epsilon_g = 5 \times 10^{-3}$ )	$0.065 \pm 0.006$	$0.349 \pm 0.010$	$0.066 \pm 0.005$	$0.351 \pm 0.010$

#### 7.7 Gate-Level QRAM Circuit Decomposition for the 4-Node Case

For the 4-node path graph  $P_4$ , the adjacency oracle decomposes into: (1) address decoding (2 Toffoli gates), (2) data loading (4 controlled- $R_y$  gates), and (3) uncomputation (2 Toffoli gates).

Total abstract gate count: 32 CNOTs + 48 single-qubit gates. Circuit depth: 18.

**Hardware-Specific Compilation.** IBM Heavy-Hex: 68 CNOTs + 48 single-qubit gates, depth  $\sim 42$ . SWAP inflation factor:  $2.1\times$ . Trapped-Ion (IonQ): no SWAP overhead, depth 18 preserved.

Table 7: Predicted vs. observed fidelities under the independent-error approximation.

Experiment	$G$	$G_1$	$G_2$	$\mathcal{F}_{\text{pred}}$	$\mathcal{F}_{\text{corr}}$	$\mathcal{F}_{\text{obs}}$
Exp 1 ( $P_4$ , identity)	96	60	36	0.618	0.830	0.94
Exp 2 ( $P_4$ , trained)	112	68	44	0.571	0.800	0.91
Exp 3 ( $C_4$ , identity)	100	62	38	0.607	0.824	0.93

The heavy-hex compilation adjusts the break-even error rate from  $\epsilon_g < 0.014$  to:

$$\epsilon_g^{\text{heavy-hex}} < \frac{\ln(17)}{2 \times 210} \approx 0.0067 \quad (13)$$

## 8 Discussion

### 8.1 Advantages of QEMA-G

Beyond quantitative complexity improvements, QEMA-G offers a modular, layered architecture enabling incremental adoption alongside existing classical infrastructure [61]. The precise interface definitions allow independent improvement of each component as quantum technology matures.

### 8.2 Limitations

**QRAM Dependence:** The most significant limitation is QEMA-G’s dependence on QRAM technology that remains experimentally immature. We have addressed the opportunity cost critique [42] quantitatively in Section 5, showing that QEMA-G survives this critique for dense and power-law graphs but not for sparse, small-degree graphs.

**Amplitude Encoding Overhead:** General amplitude encoding requires  $O(d)$  gates, which dominates the per-node cost in the realistic regime.

**Barren Plateaus:** The barren plateau problem [62] may impede VQC training at scale. For QEMA-G specifically, the risk scales with feature qubit count ( $\lceil \log_2 d \rceil$ ), which is logarithmic in the feature dimension.

**Re-encoding Cost:** The hybrid iterative scheme introduces classical re-encoding at every layer, contributing  $O(n \cdot d)$  cost per layer.

**Measurement Overhead:** Extracting classical information requires repeated measurements. Shot counts scale as  $O(d/\epsilon^2)$  naively, or  $O(\log d/\epsilon^2)$  with classical shadow estimation [60].

**Training Cost Analysis.** The parameter-shift rule requires  $2|\theta_{\text{diff}}|$  circuit executions per gradient evaluation per node, each with  $S$  measurement shots. Total training cost per epoch:

$$C_{\text{train}} = n \cdot k \cdot (2|\theta_{\text{diff}}| \cdot S \cdot D_{\text{circuit}} + C_{\text{classical}}) \quad (14)$$

**Worked example (Fraud Detection Graph).** For the payment processor graph ( $n = 10^8$ ,  $k = 2$ ,  $d = 128$ ,  $L = 2$ ): total quantum operations per epoch  $\approx 1.9 \times 10^{18}$ . A classical 2-layer GNN requires  $\sim 1.7 \times 10^{13}$  FLOPs per epoch. Training cost ratio:  $1.9 \times 10^{18}/1.7 \times 10^{13} \approx 10^5$ .

The quantum training pipeline is approximately five orders of magnitude more expensive than classical training, driven by three factors: (i) the parameter-shift rule’s  $2|\theta|$ -fold circuit repetition; (ii) shot overhead  $S$  per circuit execution; and (iii) the sequential nature of quantum gradient estimation.



**Implication:** QEMA-G’s advantage case is substantially stronger for inference-dominated workloads (real-time fraud detection, recommendation serving) than for training-dominated workloads. For applications requiring frequent model retraining, the  $\sim 10^5$  training cost ratio represents a significant practical barrier.

### 8.3 Inference Economics and Break-Even Query Volume

The  $\sim 10^5 \times$  training cost asymmetry raises a natural question: *at what inference query volume does quantum deployment become cost-effective?* We model this as a simple amortization problem.

Let  $C_{\text{train}}^{(q)} = \beta \cdot C_{\text{train}}^{(c)}$  denote the quantum training cost as a multiple  $\beta \approx 10^5$  of the classical training cost. Let  $C_{\text{query}}^{(q)} = C_{\text{query}}^{(c)} / \gamma$  denote the per-query quantum inference cost, where  $\gamma$  is the effective inference speedup (using the  $d$ -parallel ratio from Example 8.1,  $\gamma \approx 5.3$ ). The total cost of the quantum pipeline over  $Q$  inference queries is:

$$C_{\text{total}}^{(q)} = C_{\text{train}}^{(q)} + Q \cdot C_{\text{query}}^{(q)} = \beta \cdot C_{\text{train}}^{(c)} + Q \cdot \frac{C_{\text{query}}^{(c)}}{\gamma} \quad (15)$$

The quantum pipeline becomes cheaper than the classical pipeline ( $C_{\text{total}}^{(q)} < C_{\text{total}}^{(c)} = C_{\text{train}}^{(c)} + Q \cdot C_{\text{query}}^{(c)}$ ) when:

$$Q > \frac{(\beta - 1) \cdot C_{\text{train}}^{(c)}}{C_{\text{query}}^{(c)} \cdot (1 - 1/\gamma)} \approx \frac{\beta}{1 - 1/\gamma} \quad (16)$$

For  $\beta = 10^5$  and  $\gamma = 5.3$ :  $Q > 10^5 / (1 - 1/5.3) \approx 1.3 \times 10^5$  queries.

**Interpretation.** A production fraud detection system processing  $> 1.3 \times 10^5$  inference queries on a single trained model would amortize the quantum training overhead. At  $10^4$  queries per second (typical for a major payment processor), this break-even is reached in approximately 13 seconds of production operation. However, if models are retrained frequently (e.g., weekly on evolving transaction patterns), the  $10^5 \times$  training cost recurs, and break-even requires correspondingly higher query volumes between retraining cycles. For inference-dominated workloads with stable models, quantum deployment is economically viable; for training-dominated workloads with frequent retraining, classical pipelines remain more cost-effective.

### 8.4 Deployment Pathways: Bridging the Training–Inference Gap

Given the  $\sim 10^5 \times$  training cost asymmetry, practitioners must choose among three deployment strategies that trade off training efficiency against inference performance:

**Pathway 1: Classical Training, Quantum Inference.** Train the VQC parameters  $\theta$  using a classical simulator that emulates the quantum aggregation pipeline, then deploy the trained parameters on quantum hardware for inference. This avoids the parameter-shift rule overhead entirely during training. The feasibility depends on the classical simulability of the VQC at training scale: for  $\lceil \log_2 d \rceil \leq 20$  feature qubits (corresponding to  $d \leq 10^6$ ), statevector simulation on GPUs is tractable, making this the most practical near-term pathway.

**Pathway 2: One-Time Quantum Training, Amortized Inference.** Accept the  $10^5 \times$  training cost as a one-time expense and amortize over millions of inference queries. As shown in Section 8.3, break-even requires  $\sim 1.3 \times 10^5$  queries—achievable within seconds for high-throughput production systems. This pathway is viable for applications with stable models and high query volumes (e.g., persistent fraud detection rules, recommendation engines with infrequent retraining).

**Pathway 3: Hybrid Pre-training with Quantum Fine-tuning.** Pre-train a classical GNN on the full graph to obtain initial parameters, then fine-tune the VQC on a quantum device using a small number of gradient steps. This reduces the quantum training cost from  $O(E \cdot n \cdot k \cdot |\theta| \cdot S)$

to  $O(E' \cdot n \cdot k \cdot |\theta| \cdot S)$  where  $E' \ll E$  is the number of fine-tuning epochs. The classical pre-training provides a warm start in parameter space, mitigating both the training cost and the barren plateau risk [62].

**Recommendation.** Pathway 1 (classical training, quantum inference) is the most immediately actionable and aligns with QEMA-G’s strongest advantage regime: inference-dominated workloads where the per-query depth advantage (Section 4.2) translates to reduced latency for high-degree hub nodes. Pathway 3 is promising as a medium-term strategy once quantum hardware supports reliable gradient estimation at moderate scale.

## 8.5 Comparison with Existing Approaches

Note that the detailed comparison table (Table 2) was presented in Section 2.2 to contextualize QEMA-G’s architectural positioning within the landscape of quantum graph approaches.

**Impact of Recent Classical Advances on Break-Even Analysis.** Sparse attention mechanisms [25, 26] reduce the effective classical cost for high-degree nodes from  $O(d_i \cdot d)$  toward  $O(k_{\text{att}} \cdot d)$  where  $k_{\text{att}} \ll d_i$ , potentially raising the degree threshold at which QEMA-G’s logarithmic access becomes advantageous. GNN-specific hardware accelerators [29] exploit data locality to achieve higher throughput. However, neither advance eliminates the fundamental  $O(d_i)$  scaling for full-neighborhood aggregation (required for exact GCN computation), and the break-even analysis in Section 5 remains valid for workloads requiring exact rather than approximate aggregation.

## 8.6 Classical Data Structure Optimizations and Fair Baselines

Throughout this paper, the complexity comparisons use  $O(d_i)$  as the classical baseline for per-node neighborhood access, corresponding to iterating over an adjacency list. However, optimized classical graph data structures can achieve faster-than-naive adjacency lookup:

- **Hash-based adjacency:** Hash tables provide  $O(1)$  expected-time lookup for individual neighbor queries, though enumerating all  $d_i$  neighbors still requires  $O(d_i)$  time.
- **Compressed Sparse Row (CSR) format:** The standard format for sparse graph computation in libraries such as igraph and PyG, CSR provides  $O(1)$  offset lookup and  $O(d_i)$  sequential neighbor enumeration with cache-friendly memory access patterns.
- **Skip lists and B-trees:** Sorted adjacency structures provide  $O(\log d_{\text{max}})$  lookup per individual neighbor query.

Critically, GNN aggregation requires *enumerating* all neighbors (to compute the sum or mean), not merely testing membership. For this operation, all classical data structures require  $\Omega(d_i)$  time, because each neighbor’s features must be read and accumulated. QEMA-G’s  $O(\log n)$  QRAM access loads the entire neighborhood in superposition, avoiding the per-neighbor enumeration entirely—this is the fundamental structural difference that survives classical optimization.

That said, the classical baselines used throughout this paper (sequential depth  $O(d_i \cdot d)$ ,  $d$ -parallel depth  $O(d_i)$ ) assume unoptimized sequential processing. Production graph libraries (e.g., DGL, PyG on NVIDIA A100 GPUs) exploit batched sparse matrix operations, vectorized scatter-gather, and hardware prefetching to achieve substantially better constants than the asymptotic  $O(d_i \cdot d)$  bound suggests. The speedup ratios reported in this paper ( $675\times$  sequential,  $5.3\times$   $d$ -parallel) should therefore be interpreted as *upper bounds against worst-case classical implementations*, not as predictions of quantum-vs-optimized-library performance. Empirical benchmarking against production graph libraries—beyond the scope of this theoretical work—would refine these estimates.

## 8.7 Dequantization Boundaries

Tang’s dequantization framework [37] shows that quantum speedups based on amplitude encoding can sometimes be replicated classically given  $\ell^2$ -norm sampling access. For QEMA-G, this applies to the embedding compression claim. However, dequantization does not apply to: (1) the QRAM-backed adjacency oracle, which provides a fundamentally different access pattern not replicable by sampling, and (2) the Grover-based graph search, which provides a provable quadratic speedup not subject to dequantization.

## 8.8 Practical Industry Implications

While QEMA-G remains a theoretical framework requiring hardware that does not yet exist at the necessary scale, its advantage regimes (Section 4) map onto three industry sectors with graph-intensive workloads: social networks ( $10^9$ – $10^{10}$  nodes, dense hub structure), pharmaceutical drug discovery (subgraph pattern matching over molecular libraries [63]), and financial transaction graphs (extreme power-law degree distributions with  $d_{\max} > 10^4$ ).

**Example 8.1** (End-to-End Fraud Detection Walkthrough—Corrected Metrics). *Consider a payment processor with  $n = 10^8$  accounts and  $m = 10^{10}$  transactions. A fraud alert on merchant node  $v$  with  $d_v = 15,000$  daily transaction partners requires 2-hop aggregation ( $k = 2$ ) with  $d = 128$ -dimensional embeddings.*

*Classical pipeline: Layer 1 aggregates  $v$ ’s 15,000 neighbors with per-node sequential depth  $15,000 \times 128 = 1.92 \times 10^6$ .*

*QEMA-G pipeline (realistic regime): Layer 1 processes node  $v$  in  $O(\log 10^8 + 128 + 2 \times 7) = O(169)$  depth—identical to processing a degree-50 node.*

*Depth ratio for node  $v$ :  $1.92 \times 10^6 / 169 \approx 11,400\times$  in raw sequential depth.*

*Closed-form overhead accounting: (1) Measurement shot overhead  $S$ : factor  $2.7\times$ . (2) SWAP overhead  $W$ : factor  $2.5\times$ . (3) Classical re-encoding  $R$ : factor  $2.5\times$ .*

*Effective speedup (sequential classical baseline):*

$$\frac{11,400}{S \times W \times R} = \frac{11,400}{2.7 \times 2.5 \times 2.5} = \frac{11,400}{16.9} \approx 675\times \quad (17)$$

*Effective speedup ( $d$ -parallel classical baseline):*

$$\frac{88.8}{2.7 \times 2.5 \times 2.5} = \frac{88.8}{16.9} \approx 5.3\times \quad (18)$$

*Per the decision framework in Section 4.2, the  $5.3\times$   $d$ -parallel speedup is the operationally relevant metric for practitioners using GPU-accelerated classical baselines.*

The fraud detection analysis illustrates two general points: (i) the headline depth ratio ( $11,400\times$ ) and practical effective speedup ( $5$ – $675\times$ ) can diverge by orders of magnitude once overhead factors are accounted multiplicatively, and (ii) QEMA-G’s strongest practical value may be in eliminating tail-latency variance rather than improving median-case throughput, since its depth is degree-independent.

## 8.9 Honest Assessment of the Gap

Current quantum hardware cannot implement QEMA-G at a scale that provides advantage over classical GNNs [13]. The smallest graph for which idealized advantages become meaningful (approximately  $n > 10^3$  with  $d > 64$ ) requires QRAM with  $> 10^3$  routing qubits operating at fidelities well below current error rates.

We estimate that fault-tolerant QRAM capable of supporting QEMA-G at advantage-providing scales is approximately 5–10 years away, based on: (1) IBM’s quantum roadmap

projecting 100,000+ qubit systems by 2033—though we note that hardware timeline predictions in quantum computing have historically been unreliable; (2) QRAM-specific designs demonstrated in principle but not physically realized beyond  $n \sim 10$ ; and (3) the small gap between current gate fidelities ( $\sim 99.5\%$ ) and the QEMA-G break-even threshold ( $> 98.6\%$ ). The 5–10 year estimate should be interpreted as a rough order-of-magnitude projection rather than a firm prediction.

The value of this work lies in providing precise target specifications—qubit count ( $> 10^3$  routing qubits), gate fidelity ( $\epsilon_g < 0.014$  abstract,  $\epsilon_g < 0.0067$  after heavy-hex compilation), connectivity (binary tree topology), and coherence time ( $> 50 \mu s$ )—for what quantum hardware must achieve to enable graph AI applications.

## 9 Future Directions

### 9.1 Near-Term (1–3 Years)

**Experimental Validation on Real Hardware.** Scaling the toy-scale simulation to 8–16 node graphs on quantum hardware. IonQ’s trapped-ion processors are particularly promising due to all-to-all qubit connectivity.

**Benchmarking on Standard GNN Tasks.** Evaluating quantum GNN performance on standard benchmarks (Cora/CiteSeer for node classification, MUTAG/PROTEINS for graph classification) even in simulation.

**Barren Plateau Mitigation for Graph VQCs.** Investigating graph-structure-aware ansatz initialization, layer-wise training [12], and local cost functions.

### 9.2 Medium-Term (3–5 Years)

**Efficient State Preparation.** Developing amplitude encoding methods that exploit GNN feature statistics to reduce preparation cost below  $O(d)$ .

**Quantum Graph Transformers.** Extending QEMA-G’s quantum attention mechanism to global self-attention with  $O(\log n)$  depth.

**Fault-Tolerant QRAM Design.** Designing QRAM architectures requiring routing qubits proportional to  $d_{\max}$  rather than  $n$ .

### 9.3 Long-Term (5–10+ Years)

**Formal WL Characterization.** Rigorously characterizing whether quantum GNNs operating in Hilbert space can distinguish graph structures beyond the  $k$ -WL hierarchy [21, 66].

**Distributed Quantum Graph Processing.** Designing protocols for distributed quantum graph processing across networked QPUs connected by quantum communication channels [67].

## 10 Conclusion

We have presented QEMA-G, a comprehensive theoretical framework integrating quantum memory architectures with graph-based AI systems. The framework comprises a four-layer architecture formalized through quantum adjacency and feature oracles, a four-phase quantum message-passing protocol with a specified hardware-efficient VQC ansatz, and a quantum graph attention mechanism.

Our dual-regime complexity analysis—the central technical contribution—establishes that: (1) under idealized QRAM assumptions, QEMA-G achieves logarithmic-depth aggregation versus linear classical scaling; (2) under realistic conditions, the advantage narrows to a depth ratio

of approximately  $d_i$  (the node degree), persisting for dense and power-law graphs but vanishing for sparse regular graphs with  $d_i \leq \log n$ ; and (3) the QRAM opportunity cost critique is survivable for power-law and dense graph topologies.

The toy-scale Qiskit validation on 4-node subgraphs—with both path and cycle topologies, and both identity and trained VQC configurations—confirmed protocol correctness and basic learning pipeline viability, with fidelity  $\mathcal{F} = 0.91\text{--}0.97$  under realistic IBM Brisbane noise conditions. The 8-node noiseless classification experiment (Experiment 4), including comparison against a matched-parameter classical baseline, provided initial evidence that the quantum aggregation mechanism may produce features with discriminative structure beyond what classical mean-aggregation with equivalent capacity achieves, though this finding requires validation at larger scales.

The practitioner’s decision framework (Section 4.2) clarifies that the operationally meaningful speedup metric depends on deployment context, with the  $d$ -parallel ratio ( $\sim 5.3\times$  for hub nodes in fraud detection) being the most relevant for GPU-accelerated production systems. The inference economics analysis (Section 8.3) establishes that quantum deployment becomes cost-effective after  $\sim 1.3 \times 10^5$  inference queries—achievable within seconds for high-throughput systems—provided models are not retrained frequently. Three concrete deployment pathways (Section 8.4) bridge the training–inference cost asymmetry, with classical training and quantum inference being the most immediately actionable.

Equally important, we identified three quantum disadvantage regimes where QEMA-G provides no advantage, and characterized the training cost asymmetry ( $\sim 10^5$  for the parameter-shift rule vs. classical backpropagation), ensuring practitioners can assess applicability across both inference and training workloads. The comparison against optimized classical data structures (Section 8.6) confirms that reported speedup ratios represent upper bounds against worst-case classical implementations.

While full realization awaits QRAM maturation, the precise hardware specifications derived here provide concrete, actionable targets for the quantum hardware community.

## Acknowledgments

The authors thank the anonymous reviewers for their constructive feedback, which substantially improved the rigor and clarity of this work. Computational resources for quantum simulations were provided by IBM Quantum through the Qiskit open-source framework.

**Funding Statement.** This work received no external funding.

**Competing Interests.** The authors declare no competing interests.

**Data and Code Availability.** All simulation code, trained model parameters, noise model configurations, and analysis scripts are publicly available at <https://github.com/AravindB98/qemag-validation>.

## A Preliminaries and Notation

This appendix introduces all technical concepts from graph theory, machine learning, and quantum computing used throughout the paper. Readers familiar with these areas may proceed directly to the main text.

### A.1 Graph Theory Foundations

A graph  $G = (V, E)$  consists of a set of nodes  $V$  and a set of edges  $E \subseteq V \times V$ . The adjacency matrix  $A \in \{0, 1\}^{n \times n}$  has  $A_{ij} = 1$  if there is an edge between nodes  $i$  and  $j$ . The degree  $d_i$  of node  $i$  is its number of neighbors. The neighborhood  $N(i)$  is the set of all nodes directly connected to  $i$ . A power-law graph [58] has degree distribution following  $P(k) \propto k^{-\gamma}$ .

## A.2 Machine Learning on Graphs

Graph Neural Networks (GNNs) [1, 19] are neural network architectures for graph-structured data. GCNs [3] update each node’s representation via  $H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)})$ . GATs [4] extend GCNs with learned attention weights. MPNNs [5] unify GNN variants through message, aggregate, update phases. The Weisfeiler-Lehman test [21, 66] bounds GNN expressiveness. Dequantization [37] refers to the discovery that certain quantum speedups can be replicated classically given appropriate sampling access.

## A.3 Quantum Computing Fundamentals

Qubits are fundamental units of quantum information:  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  with  $|\alpha|^2 + |\beta|^2 = 1$  using Dirac notation [70]. Key quantum gates:  $R_y(\theta)$  rotates around the  $y$ -axis of the Bloch sphere;  $R_z(\theta)$  rotates around the  $z$ -axis; CNOT flips the target qubit if the control is  $|1\rangle$ .

## A.4 Quantum Computing for Graphs

A quantum oracle encodes data into quantum states. QRAM [10] enables superposition queries; the bucket-brigade [9] achieves  $O(\log N)$  depth with  $O(N)$  qubits; Fat-Tree QRAM [41] supports parallel queries. Amplitude encoding [71] maps  $d$ -dimensional vectors to  $\lceil \log_2 d \rceil$ -qubit states. Variational Quantum Circuits (VQCs) [12] are parameterized circuits optimized through classical-quantum hybrid loops. Grover’s algorithm [38] achieves  $O(\sqrt{N})$  unstructured search, provably optimal.

## A.5 NISQ Computing

NISQ [13] describes current quantum computers (50–1000+ qubits, limited by noise). The barren plateau problem [62] causes exponentially vanishing gradients in large VQCs. Error mitigation [59] reduces noise impact without full error correction. Classical shadow estimation [60] efficiently extracts quantum state properties. Dynamical decoupling [76] uses control pulses to average out environmental noise.

## References

- [1] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” *IEEE Trans. NNLS*, vol. 32, no. 1, pp. 4–24, 2021.
- [2] J. Zhou, G. Cui, S. Hu, et al., “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57–81, 2020.
- [3] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. ICLR*, 2017.
- [4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *Proc. ICLR*, 2018.
- [5] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural message passing for quantum chemistry,” in *Proc. ICML*, pp. 1263–1272, 2017.
- [6] J. Chen, T. Ma, and C. Xiao, “FastGCN: Fast learning with graph convolutional networks via importance sampling,” in *Proc. ICLR*, 2018.
- [7] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in NeurIPS*, vol. 30, pp. 1024–1034, 2017.

- [8] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, “Quantum machine learning,” *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.
- [9] V. Giovannetti, S. Lloyd, and L. Maccone, “Architectures for a quantum random access memory,” *Physical Review A*, vol. 78, no. 5, p. 052310, 2008.
- [10] V. Giovannetti, S. Lloyd, and L. Maccone, “Quantum random access memory,” *Physical Review Letters*, vol. 100, no. 16, p. 160501, 2008.
- [11] A. Peruzzo, J. McClean, P. Shadbolt, et al., “A variational eigenvalue solver on a photonic quantum processor,” *Nature Communications*, vol. 5, p. 4213, 2014.
- [12] M. Cerezo, A. Arrasmith, R. Babbush, et al., “Variational quantum algorithms,” *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.
- [13] J. Preskill, “Quantum computing in the NISQ era and beyond,” *Quantum*, vol. 2, p. 79, 2018.
- [14] F. Arute, K. Arya, R. Babbush, et al., “Quantum supremacy using a programmable superconducting processor,” *Nature*, vol. 574, pp. 505–510, 2019.
- [15] D. Vrandečić and M. Krötzsch, “Wikidata: A free collaborative knowledgebase,” *Commun. ACM*, vol. 57, no. 10, pp. 78–85, 2014.
- [16] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, “Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks,” in *Proc. ACM SIGKDD*, pp. 257–266, 2019.
- [17] D. Zheng, C. Ma, M. Wang, et al., “DistDGL: Distributed graph neural network training for billion-scale graphs,” in *Proc. IEEE/ACM BigData*, pp. 36–44, 2020.
- [18] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *J. Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.
- [19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” *IEEE Trans. Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.
- [20] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *Advances in NeurIPS*, vol. 29, pp. 3844–3852, 2016.
- [21] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, “How powerful are graph neural networks?” in *Proc. ICLR*, 2019.
- [22] V. P. Dwivedi and X. Bresson, “A generalization of transformer networks to graphs,” in *AAAI Workshop DLG-AAAI*, 2021.
- [23] A. Vaswani, N. Shazeer, N. Parmar, et al., “Attention is all you need,” in *Advances in NeurIPS*, vol. 30, pp. 5998–6008, 2017.
- [24] U. Alon and E. Yahav, “On the bottleneck of graph neural networks and its practical implications,” in *Proc. ICLR*, 2021.
- [25] J. Chen, K. Gao, G. Li, and K. He, “NAGphormer: A tokenized graph transformer for node classification in large graphs,” in *Proc. ICLR*, 2023.
- [26] Q. Wu, Z. Zhao, C. Yang, et al., “NodeFormer: A scalable graph structure learning transformer for node classification,” in *Advances in NeurIPS*, vol. 35, 2022.

- [27] W. Cong, R. Forsati, M. Kandemir, and M. Mahdavi, “Minimal variance sampling with provable guarantees for fast training of graph neural networks,” in *Proc. KDD*, pp. 1393–1403, 2020.
- [28] J. Chen, J. Zhu, and L. Song, “Stochastic training of graph convolutional networks with variance reduction,” in *Proc. ICML*, pp. 942–950, 2018.
- [29] S. Abadal, A. Jain, R. Guirado, J. López-Alonso, and E. Alarcón, “Computing graph neural networks: A survey from algorithms to accelerators,” *ACM Computing Surveys*, vol. 54, no. 9, pp. 1–38, 2021.
- [30] K. Kinningham, P. Levis, and C. Ré, “GNN hardware acceleration: A survey,” *IEEE Micro*, vol. 43, no. 4, pp. 36–44, 2023.
- [31] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” *arXiv preprint arXiv:1411.4028*, 2014.
- [32] G. Verdon, T. McCourt, E. Luzhnica, V. Singh, S. Leichenauer, and J. Hidary, “Quantum graph neural networks,” *arXiv preprint arXiv:1909.12264*, 2019.
- [33] B. Zheng, G. Li, M. Shan, Y. Fan, Y. Xie, and A. C. Lim, “QuGCN: Quantum graph convolutional neural network,” in *Proc. DAC*, 2022.
- [34] L. Bai, L. Cui, Y. Jiao, L. Rossi, and E. R. Hancock, “Learning backtrackless aligned-spatial graph convolutional networks for graph classification,” *IEEE Trans. PAMI*, vol. 44, no. 2, pp. 783–798, 2022.
- [35] A. Ceschini, F. Mauro, F. De Falco, et al., “From graphs to qubits: A critical review of quantum graph neural networks,” *arXiv preprint arXiv:2408.06524*, 2024.
- [36] N. X. Tung et al., “Scalable quantum message passing graph neural networks for next-generation wireless communications,” *arXiv preprint arXiv:2601.18198*, 2025.
- [37] E. Tang, “A quantum-inspired classical algorithm for recommendation systems,” in *Proc. ACM STOC*, pp. 217–228, 2019.
- [38] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proc. 28th ACM STOC*, pp. 212–219, 1996.
- [39] J. Kim et al., “Next-generation graph computing with electric current-based and quantum-inspired approaches,” *Nature Communications*, vol. 16, 2025.
- [40] P. Mukhopadhyay, “A quantum random access memory (QRAM) using a polynomial encoding of binary strings,” *Scientific Reports*, vol. 15, p. 11002, 2025.
- [41] S. Xu, Y. Ding, and A. Lu, “Fat-Tree QRAM: A high-bandwidth shared quantum random access memory for parallel queries,” *Proc. ASPLOS*, vol. 2, pp. 390–406, 2025.
- [42] S. Jaques and A. G. Rattew, “QRAM: A survey and critique,” *Quantum*, vol. 9, p. 1922, 2025.
- [43] P. Mernyei, K. Raissi, A. Mukherjee, and H. J. Briegel, “Equivariant quantum circuits for learning on weighted graphs,” *npj Quantum Information*, vol. 9, no. 1, p. 47, 2023.
- [44] V. V. Shende, S. S. Bullock, and I. L. Markov, “Synthesis of quantum-logic circuits,” *IEEE Trans. CAD*, vol. 25, no. 6, pp. 1000–1010, 2006.



- [45] A. Kandala, A. Mezzacapo, K. Temme, et al., “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature*, vol. 549, no. 7671, pp. 242–246, 2017.
- [46] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, 2015.
- [47] M. Schuld and N. Killoran, “Quantum machine learning in feature Hilbert spaces,” *Physical Review Letters*, vol. 122, no. 4, p. 040504, 2019.
- [48] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” in *Proc. ICML*, pp. 807–814, 2010.
- [49] D. Hendrycks and K. Gimpel, “Gaussian error linear units (GELUs),” *arXiv preprint arXiv:1606.08415*, 2016.
- [50] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proc. ICML*, pp. 448–456, 2015.
- [51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *JMLR*, vol. 15, no. 56, pp. 1929–1958, 2014.
- [52] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE CVPR*, pp. 770–778, 2016.
- [53] D. Rogers and M. Hahn, “Extended-connectivity fingerprints,” *J. Chem. Inf. Model.*, vol. 50, no. 5, pp. 742–754, 2010.
- [54] Q. Li, Z. Han, and X.-M. Wu, “Deeper insights into graph convolutional networks for semi-supervised learning,” *Proc. AAAI*, vol. 32, no. 1, 2018.
- [55] S. A. Cook, “The complexity of theorem-proving procedures,” in *Proc. 3rd ACM STOC*, pp. 151–158, 1971.
- [56] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, “The power of quantum neural networks,” *Nature Computational Science*, vol. 1, no. 6, pp. 403–409, 2021.
- [57] V. Havlíček, A. D. Córcoles, K. Temme, et al., “Supervised learning with quantum-enhanced feature spaces,” *Nature*, vol. 567, no. 7747, pp. 209–212, 2019.
- [58] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [59] K. Temme, S. Bravyi, and J. M. Gambetta, “Error mitigation for short-depth quantum circuits,” *Physical Review Letters*, vol. 119, no. 18, p. 180509, 2017.
- [60] H.-Y. Huang, R. Kueng, and J. Preskill, “Predicting many properties of a quantum system from very few measurements,” *Nature Physics*, vol. 16, no. 10, pp. 1050–1057, 2020.
- [61] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, “Geometric deep learning: Grids, groups, graphs, geodesics, and gauges,” *arXiv preprint arXiv:2104.13478*, 2021.
- [62] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, “Barren plateaus in quantum neural network training landscapes,” *Nature Communications*, vol. 9, no. 1, p. 4812, 2018.

- [63] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Message passing neural networks for molecular property prediction,” in *Machine Learning Meets Quantum Physics*, pp. 199–214, Springer, 2020.
- [64] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Advances in NeurIPS*, vol. 26, pp. 2787–2795, 2013.
- [65] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, “Embedding entities and relations for learning and inference in knowledge bases,” in *Proc. ICLR*, 2015.
- [66] B. Weisfeiler and A. Lehman, “The reduction of a graph to canonical form and the algebra which appears therein,” *NTI, Series 2*, vol. 9, pp. 12–16, 1968.
- [67] H. J. Kimble, “The quantum internet,” *Nature*, vol. 453, no. 7198, pp. 1023–1030, 2008.
- [68] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral networks and locally connected networks on graphs,” in *Proc. ICLR*, 2014.
- [69] F. R. K. Chung, *Spectral Graph Theory*, American Mathematical Society, 1997.
- [70] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge Univ. Press, 10th anniversary edition, 2010.
- [71] P. Rebentrost, M. Mohseni, and S. Lloyd, “Quantum support vector machine for big data classification,” *Physical Review Letters*, vol. 113, no. 13, p. 130503, 2014.
- [72] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, “Exponential algorithmic speedup by a quantum walk,” in *Proc. 35th ACM STOC*, pp. 59–68, 2003.
- [73] S. Dernbach, A. Mohseni-Kabir, S. Pal, and D. Towsley, “Quantum walk neural networks for graph-structured data,” in *Complex Networks VII*, pp. 182–193, Springer, 2019.
- [74] C. Zoufal, A. Lucchi, and S. Woerner, “Quantum generative adversarial networks for learning and loading random distributions,” *npj Quantum Information*, vol. 5, p. 103, 2019.
- [75] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, “Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers,” in *Proc. ASPLOS*, pp. 1015–1029, 2019.
- [76] L. Viola, E. Knill, and S. Lloyd, “Dynamical decoupling of open quantum systems,” *Physical Review Letters*, vol. 82, no. 12, pp. 2417–2421, 1999.