Quantum Algorithm for Deep Neural Networks with Efficient I/O

Anonymous authors
Paper under double-blind review

Abstract

A primary aim of research in quantum computing is the realization of quantum advantage within deep neural networks. However, it is hindered by known challenges in constructing deep architectures and the prohibitive overhead of quantum data I/O. We introduce a framework to overcome these barriers, designed to achieve an asymptotic speedup over the large input dimension of modern DNNs. This framework is based on the belief that a deep learning model can achieve similar performance when "rough" copies of the data are allowed, which is called the good-enough principle in this paper. Our framework enables the design of multi-layer Quantum ResNet and Transformer models by strategically breaking down the task into subroutines and assigning them to be executed by quantum linear algebra (QLA) or quantum arithmetic modules (QAM). This modularity is enabled by a novel data transfer protocol, Discrete Chebyshev Decomposition (DCD). Numerical validation reveals a pivotal insight: the measurement cost required to maintain a target accuracy scales sublinearly with the input dimension, verifying the good-enough principle. This sublinear scaling is key to preserving the quantum advantage, ensuring that I/O overhead does not nullify the computational gains. A rigorous resource analysis further corroborates the superiority of our models in both efficiency and flexibility. Our research provides strong evidence that quantum neural networks can be more scalable than classical counterparts on a fault-tolerant quantum computer.

1 Introduction

The current era of artificial intelligence is defined by the triumph of deep neural networks (DNNs). Large-scale models, particularly the Transformer architecture Vaswani et al. (2017), have revolutionized countless fields by leveraging immense depth to learn complex data representations. This success, however, is a double-edged sword. The computational demands of these models create a formidable bottleneck, especially for operations whose complexity scales polynomially with the primary input dimension, such as the $O(N^2)$ attention mechanism in Transformers with sequence length N. In parallel, quantum computing offers a new paradigm promising significant speedups for such tasks Nielsen & Chuang (2010); Preskill (2018). This has catalyzed research into Quantum Deep Neural Networks (QDNNs) Beer et al. (2020); Liu et al. (2024); Li et al. (2020b); Kerenidis et al. (2020); Ye et al. (2025), aiming to harness quantum mechanics to transcend the scaling limitations of classical deep learning.

However, the pursuit of practical QDNNs has splintered into two main directions, each with fundamental limitations. Variational Quantum Circuits (VQCs) Cerezo et al. (2021); Wen et al. (2024); Evans et al. (2024) are compatible with near-term hardware but generally lack provable speedups and are plagued by trainability issues like barren plateaus McClean et al. (2018); Wang et al. (2021); Anschuetz & Kiani (2022); Bittel & Kliesch (2021). Conversely, approaches based on Quantum Linear Algebra (QLA) subroutines Kerenidis & Prakash (2016); Childs et al. (2017); Liu et al. (2021); Krovi (2023); Liao & Ferrie (2024) promise demonstrable polynomial speedups. Yet, these QLA-based methods confront a critical challenge in constructing genuinely deep architectures, the quantum no-clone theory. Attempts have been made to circumvent this problem by leveraging iterative communication between classical and quantum systems Kerenidis et al. (2020). The quantum data I/O bottleneck, which theoretically bounds the overhead of faithfully reconstructing a quantum state, has largely confined these proposals to feasible constructs.

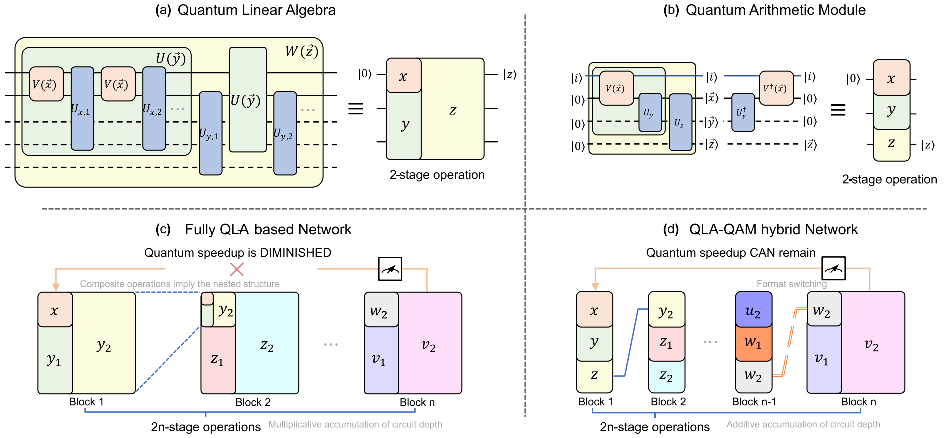

Figure 1. An overview of our hybrid quantum-classical framework for building deep quantum neural networks. (a)/(b) Typical examples of quantum linear algebra (QLA) and quantum arithmetic module (QAM) circuits and their symbolic representations. (c) The symbolic representation of existing quantum networks based on nested QLA operations, which suffer from multiplicative complexity scaling. (d) Our QLA-QAM Hybrid Network, which utilizes the additive circuit depth of QAMs to realize a practical quantum speedup.

In this paper, we address these fundamental limitations by proposing a novel quantum deep neural network framework. One core insight is to achieve practical quantum advantage by selectively accelerating only the computationally intensive parts of a DNN that bottleneck with respect to the large input dimension (N) Vaswani et al. (2017); Dao et al. (2022), while efficiently handling operations on the smaller, fixed feature dimension (d) with more flexible quantum routines. This targeted acceleration strategy leads to a hybrid quantum-classical, layer-by-layer execution model, whose design principles are illustrated in Figure 1. Our framework systematically decomposes DNN layers into three module types: Quantum Linear Algebra Modules (QLAs, green blocks) for operations that scale with the large dimension N, and Quantum Arithmetic Modules (QAMs, blue blocks) for efficient processing along the feature dimension d. The entire deep, modular architecture is made feasible by a “good-enough” information transfer principle: a deep network can achieve high performance without perfect, high-fidelity reconstruction of its intermediate states. This “good-enough” principle, inspired by the robustness of classical networks to quantization and pruning Han et al. (2015), posits that preserving salient features is more critical than exact state replication. Inspired by this observation, a novel protocol we term Discrete Chebyshev Decomposition (DCD) is proposed to improve the notorious quantum data I/O bottleneck between layers.

As depicted in Figure 1a-b, QLAs and QAMs exhibit fundamentally different scaling properties. A critical distinction lies in their composition: composing multiple QLA operations, as in existing proposals (Figure 1c), leads to a multiplicative accumulation of circuit complexity and error, often rendering theoretical speedups impractical for deep architectures. In sharp contrast, the circuit depth of our QAMs accumulates only additively. This linear scaling is crucial for constructing deep networks, enabling efficient element-wise non-linearities and parallel dot products.

This architectural choice directly addresses the viability of quantum speedup in deep networks. A fully QLA-based network (Figure 1c) struggles due to the compounding complexity of nested QLA subroutines. Our QLA-QAM hybrid model (Figure 1d), however, leverages the additive depth of QAMs to create a feasible pathway to acceleration, with the overall speedup ultimately depending on a manageable sampling cost. These modules can be flexibly assembled to form sophisticated architectures like a Quantum ResNet or a Quantum Transformer, demonstrating the framework’s versatility. The DCD protocol then acts as the crucial bridge, enabling robust inter-layer communication without prohibitive overheads.

This work presents the first concrete theoretical and empirical validation of this targeted acceleration strategy as a viable pathway toward large-scale QDNNs. Our key contributions are summarized as follows:

A Hybrid Quantum Acceleration Framework for Deep Networks: We propose a novel framework that systematically decomposes deep neural network operations. It strategically allocates large-dimension computations to QLA and small-dimension, parallel tensor operations to efficient QAMs. This design enables the construction of multi-layer Quantum ResNet and Transformer models with a provable end-to-end speedup.

Demonstrating and Exploring I/O Overhead for QDNNs: We fully take the scaling of measurement cost for quantum deep learning models into consideration. Specifically, we investigate how the measurement cost, required to maintain target accuracy, scales with the input dimension. We also introduce the Discrete Chebyshev Decomposition (DCD) protocol, a novel and efficient ”good-enough” data transfer mechanism for mitigating quantum I/O bottleneck, which demonstrates reduced dependence on system size.

Resource Analysis and Practical Advantage: Through detailed theoretical and numerical resource analysis, we quantitatively demonstrate that our hybrid approach significantly outperforms state-of-the-art fully quantum-based proposals. We further conduct a comprehensive assessment of the DCD protocol to identify the conditions under which it offers distinct advantages precisely and to quantify the extent of those benefits.

1.1 Related works

Quantum Neural Networks Research on quantum neural networks has made significant progress Zhao & Wang (2021); Valdez & Melin (2023); Peral-García et al. (2024). Levine et al. (2019) has established theoretical connections between deep learning architectures and quantum entanglement. VQCs are one major direction for quantum neural networks Mitarai et al. (2018); Cong et al. (2019); Cerezo et al. (2021). They are compatible with near-term hardware but face severe trainability issues, such as barren plateausMcClean et al. (2018); Anschuetz & Kiani (2022). However, fault-tolerant algorithms based on quantum linear algebra (QLA) can offer provable speedups Kerenidis et al. (2020); Guo et al. (2024), while both data I/O and the scaling of network depth constitute significant hurdles. Beyond these categories, alternative learning paradigms have also been explored Amin et al. (2018); Pan et al. (2023); Ye et al. (2025).

Measurement Techniques The quantum-classical I/O has been mitigated by classical preprocessing in Stein et al. (2022); Kwak et al. (2023). Novel measurement techniques have been proposed, such as shadow tomography Aaronson (2018); Huang et al. (2020), which have applications in neural networks, as seen in Abbas et al. (2023).

Quantum algorithms Quantum Arithmetic algorithms, such as quantum adders and multipliers, have been optimized over the past decades Draper (2000); Gidney (2018; 2019). Varieties of quantum Linear Algebra (QLA) algorithms have been proposed, including Harrow et al. (2009); Childs et al. (2017); Gilyén et al. (2019), and applied in machine learning tasks Lloyd et al. (2014); Kerenidis & Prakash (2016).

2 Quantum Modules

2.1 Quantum Linear Algebra (QLA)

Quantum Linear Algebra is a type of quantum algorithm that has wide applications in the field of data analysis, including notable algorithms such as quantum component analysis Lloyd et al. (2014), quantum linear system solvers Harrow et al. (2009); Wossnig et al. (2018), and quantum differential equation system solvers Berry et al. (2017); Xue et al. (2021); Liu et al. (2021).

In QLA, a matrix A is usually encoded into a quantum state by amplitude encoding Nakaji et al. (2022); Gonzalez-Conde et al. (2024):

$$|A\rangle = \frac{1}{\sqrt{\sum_{i,j} A_{ij}^2}} \sum_{i,j} A_{ij} |i\rangle |j\rangle , \tag{1}$$

or into quantum operations by block-encoding, for some constant α Wan et al. (2021):

$$(I \otimes \langle 0|)U_A(I \otimes |0\rangle) = \frac{A}{\alpha} \longleftrightarrow U_A = \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}. \tag{2}$$

The matrix multiplication AB is simple in QLA by $U_A |B\rangle$. The quantum singular value transformation even allows for a polynomial transformation $U_{p(A)}$ with polylogarithmic usage of U_A Gilyén et al. (2019). Solving problems usually requires a composite of such operations. Due to the non-clone theorem in quantum mechanics, the quantum circuit behaves as a nested structure for successive transformation, as depicted in Figure 1a, of which the overhead will accumulate multiplicatively.

2.2 Quantum Arithmetic Modules (QAMs)

Quantum Arithmetic Modules (QAMs) are the cornerstone for implementing operations on the smaller, fixed feature dimension (d) within our framework. The primary strength of QAMs lies in their ability to perform complex arithmetic operations in parallel. Given the input $|a\rangle, |b\rangle$, the function of a typical quantum adder can be written as Draper (2000); Ruiz-Perez & Garcia-Escartin (2017); Li et al. (2020a; 2021)

$$U_{\text{Add}} |a\rangle |b\rangle |0\rangle = |a\rangle |b\rangle |a+b\rangle . \tag{3}$$

The linearity and quantum superposition allow the parallel implementation:

$$U_{\text{Add}} \sum_i \alpha_i |i\rangle |a_i\rangle |b_i\rangle |0\rangle = \sum_i \alpha_i |i\rangle |a_i\rangle |b_i\rangle |a_i + b_i\rangle . \tag{4}$$

By combining quantum Adders and Multipliers, complex operations such as tensor products or contractions can be realized in parallel. Consider the operation $R_{ikjl} = \sum_\mu S_{i\mu j} T_{k\mu l}$ for tensors $S \in \mathbb{R}^{c_s \times d \times p}$ and $T \in \mathbb{R}^{c_t \times d \times q}$. The process, whose circuit structure is abstractly represented in Figure 1b, typically involves:

State Preparation: Input tensors S and T are loaded into quantum registers using controlled state preparation oracles, e.g., $O_S^{(c)} |i,j\rangle |0\rangle = |i,j\rangle \bigotimes_\mu |S_{i\mu j}\rangle$.

Parallel Computation: A series of quantum multiplier and adder circuits compute the products $S_{i\mu j} T_{k\mu l}$ for all μ in parallel and sum them into an accumulator register, resulting in the state $|i,j,k,l\rangle |R_{ikjl}\rangle$.

Uncomputation: To release ancillary qubits for reuse and maintain circuit reversibility, the inverse of the computation steps is applied to the input registers, returning them to their initial state $|0\rangle$, as visually suggested in Figure 1b.

This arithmetic-based approach allows for the efficient execution of structured linear algebra and element-wise non-linearities, providing the necessary computational primitives for deep learning layers while maintaining an additive, manageable growth in circuit complexity.

3 A Framework for Deep Quantum Networks

3.1 Intra-Layer Computation: QLAs and QAMs

Our framework's design is tailored for the common regime where a large input dimension N dominates a smaller feature dimension d (e.g., sequence length vs. embedding dimension in Transformers). Our strategy is to achieve quantum speedup specifically with respect to N. This dictates a modular separation of labor within each layer between two distinct types of quantum modules.

Figure 2 visually contrasts these two approaches.

Quantum Linear Algebra Modules (QLAs) are reserved for operations that are computationally dense and scale with the large dimension N, as shown in Figure 2a. They operate on amplitude-encoded data and utilize block-encoding algorithms to tackle the primary bottlenecks, such as the $N \times N$ matrix multiplications in Transformer attention.

Table 1: Complexity Comparison of Information Extraction Protocols for a state in $\mathbb{C}^d$.

Protocol	**Complexity**	**Classical Post-processing**	**Goal**
Full QST	$O(d^2)$	$O(d^3)$	Full density matrix reconstruction
Shadow Tomography[a]	$O(K\log(M)/\delta^2)$	$O(M \cdot \text{poly}(\log d))$	Estimate M few-body observables
DCD Protocol (Our work)	$\boldsymbol{O(r/\delta)}$	$\boldsymbol{O(r \cdot d)}$	**Extract r global feature coefficients**

[a] For estimating M observables with Pauli weight at most K to precision δ.

Quantum Arithmetic Modules (QAMs) handle computations that are sparse, element-wise, or structured along the smaller dimension d. As illustrated for the ReLU activation in Figure 2b, QAMs operate on digitally encoded numbers. They are essential for applying non-linearities (e.g., ReLU) and performing structured linear algebra where operations can be parallelized over the N items (e.g., applying a $d \times d$ weight matrix to N vectors).

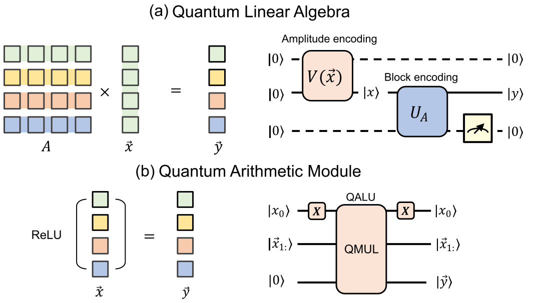

Figure 2. Mapping dense and sparse operations to corresponding quantum modules. (a) The dense lattice diagram of matrix multiplication is implemented with a QLA module. (b) The sparse, element-wise nature of the ReLU function is implemented with a QAM.

The synergy between QLAs and QAMs is the key to handling modern deep learning models: QLAs provide the speedup for large-scale, dense linear algebra, while QAMs efficiently implement the necessary non-linear activations and feature-space transformations. This combination allows for a faithful and accelerated quantum implementation of entire network layers.

3.2 Inter-Layer Communication: The Discrete Chebyshev Decomposition Protocol

The critical link in our architecture—and the concrete embodiment of our "good-enough" principle—is the protocol for information conversion between classical and quantum computer. To bypass the infeasible cost of tomography, we introduce the Discrete Chebyshev Decomposition (DCD) protocol, designed to extract a compressed classical representation of a quantum state.

Our choice of the Chebyshev basis is mathematically motivated. For any function on a finite interval, a truncated Chebyshev series provides the best polynomial approximation in the l_∞ norm (minimax approximation) Ahmed et al. (2006); Trefethen (2019). Chebyshev basis also has natural applications in QLA algorithms Martyn et al. (2021). We view the amplitudes of a quantum state $|\psi\rangle$ as evaluations of an underlying function. By projecting $|\psi\rangle$ onto the first r Chebyshev basis vectors, we find the optimal low-degree polynomial approximation of this function, capturing its most significant, low-frequency features with a minimal number of coefficients.

The DCD protocol assumes that the information in a layer's output state $|\psi\rangle$ is highly compressible. Any such state can be formally expanded in the discrete Chebyshev basis $\{|T_j\rangle\}$ as $|\psi\rangle = \sum_{j=0}^{d-1} c_j |T_j\rangle$, where $c_j = \langle T_j|\psi\rangle$. Our core hypothesis is that an approximation using only the first $r \ll d$ coefficients is sufficient for the next layer. The protocol is detailed in Algorithm 1.

Theorem 3.1. *(Discrete Chebyshev Decomposition) Given access to a state preparation unitary for $|\psi\rangle$ with cost C_ψ, the DCD protocol can estimate the first r Chebyshev coefficients to precision δ with total query complexity $\tilde{O}(r \cdot C_\psi/\delta)$. The subsequent state re-preparation for the next layer requires $O(r)$ digital encoding input and a QAM with complexity $\tilde{O}(r \cdot poly(\log d))$.*

The efficiency of the DCD protocol is its main advantage. As summarized in Table 1, DCD offers a clear advantage over the intractable scaling of QST. While shadow tomography is effective for estimating local observables, DCD is purpose-built to extract a global, spectral representation of the state. Its query complexity scales only with the desired number of features, r, and precision, δ.

Algorithm 1: Discrete Chebyshev Decomposition (DCD) Protocol

```
Input: Output state of layer $k$, $|\psi_{\text{out}}^{(k)}\rangle$; truncation rank $r$; target precision $\delta$.
Output: Classical coefficient vector $\mathbf{c}_{\text{classical}} = [c_0, c_1, \dots, c_{r-1}]^T$.
/* Coefficient Estimation */
for $j \leftarrow 0$ to $r-1$ do
    Efficiently prepare the basis state $|T_j\rangle$ using its known recurrence relation;
    Construct a circuit to project $|\psi_{\text{out}}^{(k)}\rangle$ onto $|T_j\rangle$;
    Use Quantum Amplitude Estimation (QAE) to estimate the coefficient $c_j = \langle T_j|\psi_{\text{out}}^{(k)}\rangle$ to
      precision $\delta$;
    Store the estimated real value $c_j$ classically;
end
/* State Re-preparation */
Load the classical vector $\mathbf{c}_{\text{classical}}$ into quantum digital encoding;
Use a QAM to compute the amplitudes of the approximate vector $\tilde{\psi}_i = \sum_{j=0}^{r-1} c_j T_{ji}$ for each
  computational basis state $|i\rangle$;
Prepare the input state for layer $k+1$, $|\psi_{\text{in}}^{(k+1)}\rangle = \sum_i \tilde{\psi}_i\,|i\rangle$, using a standard state preparation
  routine.
```

Since our work demonstrates that r can be significantly smaller than d, DCD transforms data transfer from an insurmountable bottleneck into a manageable subroutine, making an end-to-end quantum speedup for deep learning finally achievable.

3.3 Quantum Model Instances: qResNet & qTransformer

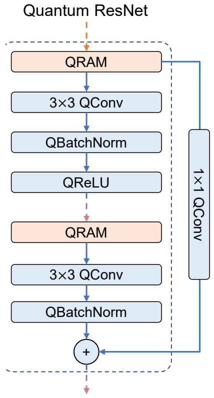

Figure 3. The quantum realization of ResNet.

To demonstrate the versatility of our framework, we now instantiate it by constructing a quantum Residual Network (qResNet) and a quantum Transformer (qTransformer). These examples showcase how our modular approach maps classical computational patterns onto the most suitable quantum primitives, guided by the principle of targeting speedups relative to the primary input dimension (e.g., sequence length N or image size $H \times W$). The concrete quantum implementations of them can be found in Appendix B.2 and B.3, together with the proof of corresponding theorems.

Quantum ResNet (qResNet) Our Quantum ResNet (qResNet) adapts the architecture of a classical ResNet-18 He et al. (2016), as depicted in Figure 3. We focus on the regime where the image's spatial dimensions (H, W) are significantly larger than the channel dimension (C). The core computational tasks are Convolution, Activation, and Residual Connections, which are implemented using the QAM A key design choice is the use of our Data Transfer Module (DTM) after each QAM-based layer. This prevents the composition of multiple sparse operators from creating a dense, computationally complex transformation, thereby preserving the efficiency of the QAM throughout the network's depth. The complexity is summarized in Theorem 3.2.

Theorem 3.2. *(Quantum ResNet Block) Given a ResNet Block, whose input tensor and kernel have shapes of (B, C, H, W) and (C, C, K, K) respectively, a quantum implementation of the ResNet Block has the quantum overhead of $\tilde{O}(CK^2 \times S(B, C, H, W))$, where $S(B, C, H, W)$ is the sampling overhead of a quantum state with shape of (B, C, H, W). The number of queries to the input data and kernel is twice for each implementation.*

Quantum Transformer (qTransformer) For vision tasks, we implement an encoder-only Quantum Transformer, focusing on the common $N \gg d$ regime, where N is the sequence length and d is the embedding dimension. This assumption dictates the allocation of tasks to create a highly efficient quantum analog, where the Feed-Forward Network and Residuals are implemented solely by QAM. Multi-Head Self-Attention(MHSA) has a hybrid implementation of QLA and QAM. This modular design is shown in Figure 4, where the blue, green, or orange blocks represent QAM, QLA, or DTM, respectively.

The complexity of a quantum Encoder Block is given in Theorem 3.3

Theorem 3.3. *(Quantum Encoder Block) For an input tensor with shape of* (B, N, d)*, where* N *is the number of tokens and* d *is the token length, the Quantum overhead of the Quantum Encoder Block is* $\tilde{O}(d^2 \times S(B, N, d))$*. The number of queries to the input data* X *is 6 for each implementation.*

The flowchart of the MHSA module (Figure 4b) showcases this strategic division of labor. This deliberate allocation of computational tasks—reserving the QLA for the true N-dimensional bottlenecks and using the QAM for structured, d-dimensional arithmetic—is the key to achieving a significant asymptotic speedup with respect to the input sequence length.

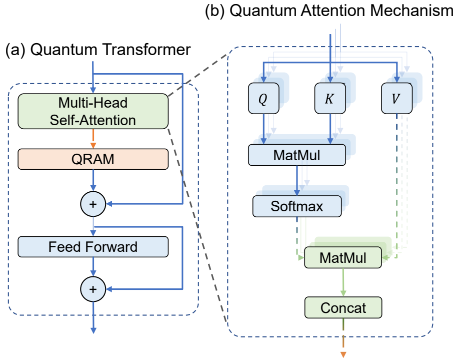

Figure 4. The quantum realization of ResNet. The red, blue, and green blocks represent DTM, QAM, and QLA, respectively

4 Experiments

This section presents numerical experiments designed to validate the efficacy and advantages of our proposed quantum-classical hybrid framework. We begin by demonstrating the superior efficiency and favorable scaling properties of our Discrete Chebyshev Decomposition (DCD) protocol when contrasted with a standard l_∞-norm tomography baseline Kerenidis et al. (2020). Subsequently, we delve into a detailed resource analysis of Quantum ResNet (qResNet) and Quantum Transformer (qTransformer) architectures, aiming to quantitatively establish the practical benefits of our integrated approach on well-established image classification benchmarks. In our experiments, the datasets we mainly use are CUB-200-2011 Wah et al. (2011), where the input dimension N represents the sequence length for Transformers and the spatial pixel count $H \times W$ for ResNet.

In our experiments, the quantum models are simulated on classical hardware. During the training phase, we implement a hybrid procedure: the forward pass simulates the quantum inference process, explicitly incorporating the sampling noise introduced by different measurement protocols to ensure the model adapts to the approximation errors. Parameter optimization (backward pass), however, is computed via standard classical backpropagation. The reported performance metrics are then evaluated using this quantum-simulated inference on the test set.

4.1 Empirical Validation of DCD Efficiency and Scaling

Figure 5 provides empirical validation of our DCD protocol's performance against the l_∞ method for both qResNet (top row, **a-c**) and qTransformer (bottom row, **d-f**). For each data transfer method, our initial step involves identifying the minimum hyperparameter configuration (specifically, sampling precision for l_∞ and rank r for DCD) necessary to achieve at least 95% of the classical model's peak performance (Figure 5a, b, d, e). Following this, we illustrate the total quantum overhead Q associated with these optimized settings as a function of the input dimension N (Figure 5c, f).

For qResNet, while both data transfer methods successfully attain the predefined target accuracy, their associated resource costs exhibit a significant divergence. As clearly

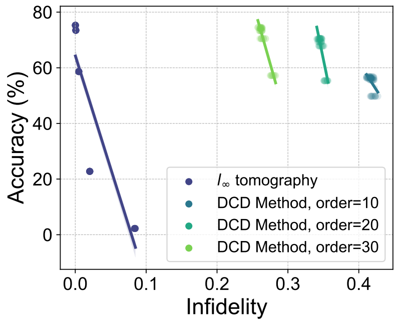

Figure 6. The relationship between classification accuracy and the infidelity of quantum state re-preparation.

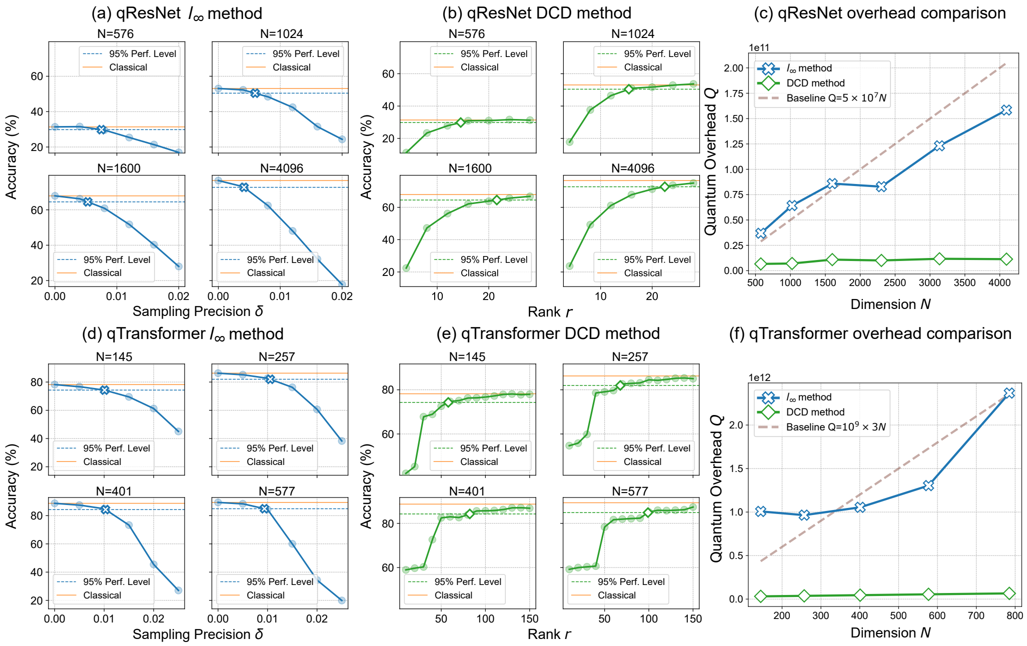

Figure 5. (a)/(d) Classification accuracy of the l_∞ tomography method for quantum ResNet-18 and Transformer across varying input dimensions N. The blue dashed line indicates the 95% performance threshold relative to the classical model, highlighted by the orange line. (b)/(e) Classification accuracy of the DCD method for both models. (c)/(f) Relationship between computed quantum resources and corresponding input dimensions N for both methods, with a linear baseline plotted for comparison.

depicted in Figure 5c, the quantum overhead for the l_∞ method scales approximately linearly with the input dimension N. The DCD overhead remains low, demonstrating its superior scaling properties for qResNet.

This inherent advantage of DCD becomes even more pronounced when applied to the qTransformer architecture. The DCD rank analysis, presented in Figure 5e, reveals a distinct ”elbow” effect, characterized by a sharp initial increase in accuracy followed by a clear plateau. This observation strongly suggests that DCD is highly effective at identifying and leveraging a compact, yet information-rich, subspace within the qTransformer's learned representations. This intrinsic efficiency directly translates into a dramatic reduction in quantum overhead, with DCD's computational cost scaling sublinearly with N (Figure 5f). This empirically observed sublinear scaling of measurement cost stands as a central and pivotal result of our study, providing compelling evidence that our DCD protocol significantly enhances the scalability of quantum deep learning by effectively mitigating the notorious I/O bottleneck.

4.2 Analysis of Quantum Properties and Hybridization

To gain deeper insights into how intrinsic quantum properties and architectural choices influence the performance of our quantum deep learning models, particularly qResNet and qTransformer, we conducted two key analyses.

First, we thoroughly investigated the impact of quantum state re-preparation infidelity on model performance across different Data Transfer Method (DTM) protocols. As illustrated in Figure 6, the DCD method consistently achieves comparable classification accuracy even with significantly lower state infidelity compared to the l_∞ method. This compelling observation robustly validates the ’good-enough’ principle in the context of quantum deep learning. It suggests that a degree of redundancy inherently exists in the data transmission pathways of deep neural networks, opening up substantial potential for quantum models to demonstrate performance advantages even under imperfect state preparation, by efficiently capturing the most salient features.

Second, we explored the performance evolution of our quantum Transformer model under varying degrees of quantum influence. This was achieved by gradually increasing the ”quantumness” of the model, specifically by progressively replacing classical layers with their quantum counterparts. Figure 7 strikingly demonstrates that this ”semi-quantum” or hybrid model effectively preserves a substantial portion of its performance, particularly during the initial stages of ”quantization.” This robustness is especially evident when employing the DCD method for data transfer, highlighting its resilience to partial quantum integration and its potential for practical, near-term hybrid implementations. This analysis underscores the flexibility and potential for incremental adoption of quantum components within classical architectures.

4.3 Comprehensive Numerical Resource Analysis

To further quantitatively substantiate the advantages of our proposed framework, we present a detailed and concrete resource analysis. Table 2a provides a direct comparison between our quantum-classical hybrid framework and a prior, fully Quantum Linear Algebra (QLA)-based model Kerenidis et al. (2020), as well as an intra-framework comparison between the l_∞ tomography and our DCD protocol. For clarity, we highlight in bold the outcomes that signify the most efficient resource utilization while maintaining equivalent or even marginally superior performance. The results demonstrate that our hybrid model achieves notably superior performance at a significantly reduced computational cost compared to the fully QLA-based model, thereby strongly confirming the inherent efficiency and architectural advantages of our design.

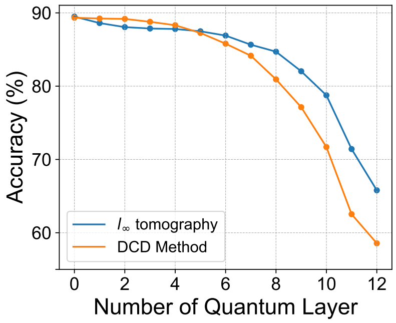

Figure 7. The relationship between model performance and the number of quantum layers within the hybrid Transformer architecture.

Furthermore, within Table 2, we conduct a thorough analysis of the critical trade-off between the two distinct data transfer methods employed within our framework. We compare the classification accuracy and the associated quantum resource cost for both qResNet and qTransformer when utilizing either the l_∞-tomography or our proposed DCD protocol. The empirical findings robustly indicate that for most of the given range of desired performance, the DCD protocol consistently offers a substantial and compelling resource advantage. While l_∞-tomography might achieve a marginally higher peak accuracy, it invariably incurs this at a disproportionately greater quantum cost. This stark contrast emphatically underscores DCD’s superior performance-to-cost ratio, making it an exceptionally attractive choice, especially in scenarios where quantum resources are inherently constrained. It also suggests that while DCD offers significant gains, these returns may diminish as one pushes towards the absolute theoretical performance limits of the model. Collectively, these comprehensive results provide direct and robust numerical evidence for the practical efficiency, scalability, and overall efficacy of our integrated quantum-classical hybrid framework.

5 Discussion

We presented a hybrid quantum-classical framework tackling the scalability challenges in quantum deep learning. Central to this design is the Discrete Chebyshev Decomposition (DCD) protocol, which alleviates the quantum I/O bottleneck and enables favorable scaling in resource overhead while maintaining improved fidelity. Resource analysis indicates that under efficient I/O mitigation, quantum advantage in deep neural networks is within reach, paving the way for more capable architectures.

This framework benefits from the compressibility of intermediate states, in line with spectral bias in classical deep learning Rahaman et al. (2019). However, its effectiveness may diminish in domains dominated by high-entropy or high-frequency features, requiring larger truncation ranks and

Table 2: Comparison of DCD and l_∞ Tomography for quantum ResNet and Transformer.

(a) Quantum ResNet

Model	Rank	Sampling Precision									
		0.002		0.004		0.010		0.020		0.040	
		Accuracy ↑ (%)	Overhead ↓ ($\times 10^9$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^9$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^9$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^9$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^9$)
DCD	10	56.44	20.44	56.80	10.23	56.23	4.10	55.68	2.06	49.78	1.03
	20	70.33	81.76	70.54	40.91	69.30	**16.39**	67.85	**8.22**	55.35	4.14
	30	74.16	183.95	74.49	**92.04**	73.32	**36.87**	70.47	**18.50**	57.21	9.31
l_∞ Tomo.	–	75.27	**989.56**	73.47	247.39	58.65	39.58	22.76	9.90	2.24	2.47
	M	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{15}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{15}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{15}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{15}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{15}$)
QLA Model	10^3	69.23	2.87	66.53	0.72	52.64	0.15	20.19	0.03	1.74	0.01
	10^5	74.77	287.24	73.44	71.81	59.39	11.49	22.99	2.87	1.47	0.72

(b) Quantum Transformer

Model	Rank	Sampling Precision									
		0.0002		0.0004		0.0010		0.0020		0.0040	
		Accuracy ↑ (%)	Overhead ↓ ($\times 10^{11}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{11}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{11}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{11}$)	Accuracy ↑ (%)	Overhead ↓ ($\times 10^{11}$)
DCD	40	59.23	5.25	58.34	2.54	52.35	0.95	44.60	0.46	31.12	0.22
	60	81.76	7.88	81.91	3.81	80.32	1.42	76.01	**0.69**	57.51	0.33
	150	86.81	**19.69**	86.78	**9.53**	86.50	**3.56**	84.86	**1.72**	78.05	**0.83**
		0.0050		0.0100		0.0150		0.0200		0.0250	
l_∞ Tomo	–	88.38	**45.82**	84.38	11.45	60.03	5.09	34.36	2.86	19.92	1.83

resulting in reduced asymptotic speedups. Quantum-inspired classical baselines remain valuable for benchmarking, though their precision scaling is less favorable for deep architectures compared to our complexity Tang (2019); Arrazola et al. (2019); Tang (2021); Chia et al. (2022).

Future work will focus on: identifying optimal I/O protocols beyond DCD, rigorously characterizing the applicability boundaries of the low-rank assumption, and exploring model design strategies that jointly achieve quantum acceleration and maintain—or surpass—classical performance while meeting re-preparation fidelity requirements.

REFERENCES

Scott Aaronson. Shadow tomography of quantum states. In *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*, pp. 325–338, 2018.

Amira Abbas, Robbie King, Hsin-Yuan Huang, William J Huggins, Ramis Movassagh, Dar Gilboa, and Jarrod R McClean. On quantum backpropagation, information reuse, and cheating measurement collapse. *arXiv preprint arXiv:2305.13362*, 2023.

Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 2006.

Mohammad H Amin, Evgeny Andriyash, Jason Rolfe, Bohdan Kulchytskyy, and Roger Melko. Quantum boltzmann machine. *Physical Review X*, 8(2):021050, 2018.

Eric R Anschuetz and Bobak T Kiani. Quantum variational algorithms are swamped with traps. *Nature Communications*, 13(1):7760, 2022.

Juan Miguel Arrazola, Alain Delgado, Bhaskar Roy Bardhan, and Seth Lloyd. Quantum-inspired algorithms in practice. *arXiv preprint arXiv:1905.10415*, 2019.

Kerstin Beer, Dmytro Bondarenko, Terry Farrelly, Tobias J Osborne, Robert Salzmann, Daniel Scheiermann, and Ramona Wolf. Training deep quantum neural networks. *Nature communications*, 11(1):808, 2020.

Dominic W Berry, Andrew M Childs, Aaron Ostrander, and Guoming Wang. Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Communications in Mathematical Physics*, 356:1057–1081, 2017.

Lennart Bittel and Martin Kliesch. Training variational quantum algorithms is np-hard. *Physical review letters*, 127(12):120502, 2021.

Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.

Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

Nai-Hui Chia, András Pal Gilyén, Tongyang Li, Han-Hsuan Lin, Ewin Tang, and Chunhao Wang. Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning. *Journal of the ACM*, 69(5):1–72, 2022.

Andrew M Childs, Robin Kothari, and Rolando D Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017.

Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Thomas G Draper. Addition on a quantum computer. *arXiv preprint quant-ph/0008033*, 2000.

Ethan N Evans, Matthew Cook, Zachary P Bradshaw, and Margarite L LaBorde. Learning with sasquatch: a novel variational quantum transformer architecture with kernel-based self-attention. *arXiv preprint arXiv:2403.14753*, 2024.

Craig Gidney. Halving the cost of quantum addition. *Quantum*, 2:74, 2018.

Craig Gidney. Windowed quantum arithmetic. *arXiv preprint arXiv:1905.07682*, 2019.

András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 193–204, 2019.

Javier Gonzalez-Conde, Thomas W Watts, Pablo Rodriguez-Grasa, and Mikel Sanz. Efficient quantum amplitude encoding of polynomial functions. *Quantum*, 8:1297, 2024.

Naixu Guo, Zhan Yu, Aman Agrawal, and Patrick Rebentrost. Quantum linear algebra is all you need for transformer architectures. *arXiv preprint arXiv:2402.16714*, 2024.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hsin-Yuan Huang, Richard Kueng, and John Preskill. Predicting many properties of a quantum system from very few measurements. *Nature Physics*, 16(10):1050–1057, 2020.

Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems, 2016.

Iordanis Kerenidis, Jonas Landman, and Anupam Prakash. Quantum algorithms for deep convolutional neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hygab1rKDS.

Andreas Klappenecker and Martin Rotteler. Discrete cosine transforms on quantum computers. In *ISPA 2001. Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis. In conjunction with 23rd International Conference on Information Technology Interfaces (IEEE Cat.*, pp. 464–468. IEEE, 2001.

Hari Krovi. Improved quantum algorithms for linear and nonlinear differential equations. *Quantum*, 7:913, 2023.

Yunseok Kwak, Won Joon Yun, Jae Pyoung Kim, Hyunhee Cho, Jihong Park, Minseok Choi, Soyi Jung, and Joongheon Kim. Quantum distributed deep learning architectures: Models, discussions, and applications. *ICT Express*, 9(3):486–491, 2023.

Yoav Levine, Or Sharir, Nadav Cohen, and Amnon Shashua. Quantum entanglement in deep learning architectures. *Physical review letters*, 122(6):065301, 2019.

Hai-Sheng Li, Ping Fan, Haiying Xia, Huiling Peng, and Gui-Lu Long. Efficient quantum arithmetic operation circuits for quantum image processing. *Science China Physics, Mechanics & Astronomy*, 63(8):280311, 2020a.

Hong Li, Nan Jiang, Zichen Wang, Jian Wang, and Rigui Zhou. Quantum matrix multiplier. *International Journal of Theoretical Physics*, 60(6):2037–2048, 2021.

YaoChong Li, Ri-Gui Zhou, RuiQing Xu, WenWen Hu, and Ping Fan. Quantum algorithm for the nonlinear dimensionality reduction with arbitrary kernel. *Quantum Science and Technology*, 6(1): 014001, 2020b.

Yidong Liao and Chris Ferrie. Gpt on a quantum computer. *arXiv preprint arXiv:2403.09418*, 2024.

Jin-Peng Liu, Herman Øie Kolden, Hari K Krovi, Nuno F Loureiro, Konstantina Trivisa, and Andrew M Childs. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proceedings of the National Academy of Sciences*, 118(35):e2026805118, 2021.

Junyu Liu, Minzhao Liu, Jin-Peng Liu, Ziyu Ye, Yunfei Wang, Yuri Alexeev, Jens Eisert, and Liang Jiang. Towards provably efficient quantum algorithms for large-scale machine-learning models. *Nature Communications*, 15(1):434, 2024.

Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature physics*, 10(9):631–633, 2014.

John M Martyn, Zane M Rossi, Andrew K Tan, and Isaac L Chuang. Grand unification of quantum algorithms. *PRX Quantum*, 2(4):040203, 2021.

Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):4812, 2018.

Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. Quantum circuit learning. *Physical Review A*, 98(3):032309, 2018.

Kosuke Mitarai, Masahiro Kitagawa, and Keisuke Fujii. Quantum analog-digital conversion. *Physical Review A*, 99(1):012301, 2019.

Kouhei Nakaji, Shumpei Uno, Yohichi Suzuki, Rudy Raymond, Tamiya Onodera, Tomoki Tanaka, Hiroyuki Tezuka, Naoki Mitsuda, and Naoki Yamamoto. Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicators. *Physical Review Research*, 4(2):023136, 2022.

Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

Xiaoxuan Pan, Zhide Lu, Weiting Wang, Ziyue Hua, Yifang Xu, Weikang Li, Weizhou Cai, Xuegang Li, Haiyan Wang, Yi-Pu Song, et al. Deep quantum neural networks on a superconducting processor. *Nature Communications*, 14(1):4006, 2023.

David Peral-García, Juan Cruz-Benito, and Francisco José García-Peñalvo. Systematic literature review: Quantum machine learning and its applications. *Computer Science Review*, 51:100619, 2024.

John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International conference on machine learning*, pp. 5301–5310. PMLR, 2019.

Lidia Ruiz-Perez and Juan Carlos Garcia-Escartin. Quantum arithmetic with the quantum fourier transform. *Quantum Information Processing*, 16(6):152, 2017.

Samuel A Stein, Betis Baheri, Daniel Chen, Ying Mao, Qiang Guan, Ang Li, Shuai Xu, and Caiwen Ding. Quclassi: A hybrid deep neural network architecture based on quantum state fidelity. *Proceedings of Machine Learning and Systems*, 4:251–264, 2022.

Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st annual ACM SIGACT symposium on theory of computing*, pp. 217–228, 2019.

Ewin Tang. Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. *Physical Review Letters*, 127(6):060503, 2021.

Lloyd N Trefethen. *Approximation theory and approximation practice, extended edition*. SIAM, 2019.

Fevrier Valdez and Patricia Melin. A review on quantum computing and deep learning algorithms and their applications. *Soft Computing*, 27(18):13217–13236, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vlatko Vedral, Adriano Barenco, and Artur Ekert. Quantum networks for elementary arithmetic operations. *Physical Review A*, 54(1):147, 1996.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011.

Lin-Chun Wan, Chao-Hua Yu, Shi-Jie Pan, Su-Juan Qin, Fei Gao, and Qiao-Yan Wen. Block-encoding-based quantum algorithm for linear systems with displacement structures. *Physical Review A*, 104(6):062414, 2021.

Samson Wang, Enrico Fontana, Marco Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J Coles. Noise-induced barren plateaus in variational quantum algorithms. *Nature communications*, 12(1):6961, 2021.

Jingwei Wen, Zhiguo Huang, Dunbo Cai, and Ling Qian. Enhancing the expressivity of quantum neural networks with residual connections. *Communications Physics*, 7(1):220, 2024.

Leonard Wossnig, Zhikuan Zhao, and Anupam Prakash. Quantum linear system algorithm for dense matrices. *Physical review letters*, 120(5):050502, 2018.

Hao Xiong, Yehui Tang, Xinyu Ye, and Junchi Yan. Circuit design and efficient simulation of quantum inner product and empirical studies of its effect on near-term hybrid quantum-classic machine learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26162–26170, 2024.

Cheng Xue, Yu-Chun Wu, and Guo-Ping Guo. Quantum homotopy perturbation method for nonlinear dissipative ordinary differential equations. *New Journal of Physics*, 23(12):123035, 2021.

Qi Ye, Shuangyue Geng, Zizhao Han, Weikang Li, L-M Duan, and Dong-Ling Deng. Quantum automated learning with provable and explainable trainability. *arXiv preprint arXiv:2502.05264*, 2025.

Renxin Zhao and Shi Wang. A review of quantum neural networks: methods, models, dilemma. *arXiv preprint arXiv:2109.01840*, 2021.

A Preliminaries

To construct our framework for deep quantum networks, we leverage advanced algorithms from quantum linear algebra and quantum arithmetic, applying them to emulate classical architectures like ResNet and the Transformer. This section aims to review these fundamental building blocks.

A.1 Quantum Subroutines for Linear Algebra

Quantum Linear Algebra (QLA) promises significant speedups for classically intractable tasks, forming the computational core of many quantum machine learning proposals. While early algorithms like HHL demonstrated the potential for exponential advantage, modern QLA has largely converged around more versatile and robust techniques.

A central concept in modern QLA is block-encoding, a method for embedding a non-unitary matrix A into a larger unitary matrix U_A. Specifically, an (α, a, δ)-block-encoding of A is a unitary U_A such that

$$(\langle 0|^{\otimes a} \otimes I) U_A (|0\rangle^{\otimes a} \otimes I) = A/\alpha, \tag{5}$$

where $\alpha \geq \|A\|$ is a normalization factor, a is the number of ancillary qubits, and the approximation is up to an error δ. This technique transforms the problem of applying a matrix into the problem of implementing a unitary circuit, making it amenable to quantum computation. Many efficient block-encoding methods exist for structured matrices, such as sparse or low-rank matrices.

Once a matrix is block-encoded, its properties can be manipulated. For instance, the Quantum Singular Value Transformation (QSVT) Gilyén et al. (2019) provides a unified framework for applying polynomial functions of a matrix's singular values to a quantum state. While QSVT is a powerful and general tool, many QLA tasks, including those in our work, can be realized using a more fundamental subroutine: Quantum Amplitude Estimation (QAE) Brassard et al. (2002). QAE allows for the estimation of the amplitude of a specific basis state in a quantum superposition. For example, if a quantum state $|\psi\rangle$ is prepared such that the probability of measuring a target state $|0\rangle$ is $p = |\langle 0|\psi\rangle|^2$, QAE can estimate p with an error δ using $O(1/\delta)$ queries to the state preparation circuit, achieving a quadratic speedup over classical sampling. This subroutine is crucial for extracting information from a quantum system, such as computing the inner product between two states or the expected value of an observable.

A.2 Quantum Arithmetic for Sparse and Element-wise Operations

While QLA excels at large-scale, dense matrix operations, DNNs also rely heavily on element-wise operations, such as adding biases, applying activation functions, and executing sparse transformations. These tasks necessitate Quantum Arithmetic (QA), which performs computations directly on the numerical values encoded in quantum registers, typically using a fixed-point binary representation.

QA circuits for fundamental operations like addition and multiplication have been well-established Vedral et al. (1996); Draper (2000), with resource costs (e.g., gate count and circuit depth) scaling polynomially with the precision (number of bits) of the encoded numbers. Furthermore, by adapting logic from classical circuits, quantum computers can efficiently perform general-purpose arithmetic operations.

Corollary A.1. *Given a function $f(x) : \mathbb{R} \to \mathbb{R}$ that can be efficiently estimated classically, there exists a quantum algorithm to implement quantum arithmetic $|x\rangle|0\rangle \to |x\rangle|\tilde{f}(x)\rangle$, where $|\tilde{f}(x) - f(x)| \leq \delta$ and δ represents the computing accuracy. The gate complexity of the algorithm is $O(\mathrm{polylog}(1/\delta))$.*

Proof. As discussed in Nielsen & Chuang (2010), quantum computing can efficiently simulate classical logic circuits using reversible gates. Therefore, any function efficiently computable classically is also efficiently computable on a quantum computer. □

To exemplify the implementation of tensor operations, we assume the typical data $X \in \mathbb{R}^{a\times s}$ is digitally encoded by a quantum circuit $O_X^{(c)}$, whose output is the direct product of quantum bit strings of each component:

$$O_X^{(c)}\left|i\right\rangle\left|0\right\rangle = \left|i\right\rangle \bigotimes_j \left|X_{ij}\right\rangle . \tag{6}$$

A matrix-vector multiplication can be realized directly between X and $v \in \mathbb{R}^s$ with this encoding and quantum arithmetic. Using quantum adders and multipliers, we can achieve step by step:

$$\begin{aligned}
\left|i\right\rangle\left|0\right\rangle\left|0\right\rangle\left|0\right\rangle &\xrightarrow{\text{Data Loading}} \left|i\right\rangle \bigotimes_j \left|X_{ij}\right\rangle \bigotimes_k \left|v_k\right\rangle\left|0\right\rangle \\
&\xrightarrow{\text{Element wise multiplication}} \left|i\right\rangle \Big(\bigotimes_j \left|X_{ij}\right\rangle\left|v_j\right\rangle\Big)\left|0\right\rangle \\
&\xrightarrow{\text{Quantum adder}} \left|i\right\rangle \Big(\bigotimes_{j\neq 0} \left|X_{ij}\right\rangle\left|v_j\right\rangle\Big)\left|X_{i0}, v_0\right\rangle\left|0 + X_{i0}v_0\right\rangle \\
&\xrightarrow{\text{Step-by-step addition}} \left|i\right\rangle \Big(\bigotimes_{j\neq 1} \left|X_{ij}\right\rangle\left|v_j\right\rangle\Big)\left|X_{i1}, v_1\right\rangle\left|0 + X_{i0}v_0 + X_{i1}v_1\right\rangle \\
&\xrightarrow{s\text{ steps}} \left|i\right\rangle \Big(\bigotimes_j \left|X_{ij}\right\rangle\left|v_j\right\rangle\Big)\Big|\sum_k X_{ik}v_k\Big\rangle .
\end{aligned} \tag{7}$$

The overall gate count, regardless of the data loading subprocedure, is then $O(s)$, which is independent of a. Analysis of other sparse operations is similar. While this method is slower than the highly parallelized approach of QLA for dense matrices, its complexity scales with the number of involved elements, making it an efficient choice for sparse problems. Furthermore, QA is the primary tool for implementing non-linear activation functions, typically by computing a piecewise polynomial approximation of the target function (e.g., ReLU), which involves a sequence of arithmetic comparisons and calculations similar to classical implementations.

A.3 Classical Architectures of Interest

Our work focuses on developing quantum counterparts for two of the most influential DNN architectures.

The Transformer Vaswani et al. (2017) has become the de facto standard for sequence modeling tasks. Its core innovation is the self-attention mechanism, defined as $\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$. The primary computational bottleneck is the matrix multiplication QK^T, which scales as $O(N^2)$ with the sequence length N, making it a prime target for quantum acceleration via QLA.

The Residual Network (ResNet) He et al. (2016) introduced the concept of residual connections, $y = \mathcal{F}(x) + x$, where $\mathcal{F}(x)$ is a block of layers. This "shortcut" structure effectively mitigates the vanishing gradient problem, enabling the training of networks with hundreds or even thousands of layers. Quantum analogues of ResNet provide an ideal testbed for assessing the ability of a quantum framework to handle truly deep architectures.

A.4 Quantum Random Access Memory and Resource Trade-offs

The practical implementation of the quantum input model relies on Quantum Random Access Memory (QRAM) to load classical data vectors into quantum states. Specifically, for a classical dataset $B = \{b_j\}_{j=0}^{N-1}$, the QRAM operation performs the mapping:

$$\sum_{j=0}^{N-1} \alpha_j |j\rangle|0\rangle \xrightarrow{\text{QRAM}} \sum_{j=0}^{N-1} \alpha_j |j\rangle|b_j\rangle, \tag{8}$$

where $|j\rangle$ is the address register and $|b_j\rangle$ is the data register.

A detailed resource analysis by Clader et al. (2022) highlights that the cost of QRAM is not negligible and presents distinct trade-offs between circuit depth (runtime) and gate count (hardware size). They analyze two primary architectures: the Bucket-Brigade (BB) model, which is optimized for noise resilience, and the Select-Swap (SS) model, which offers tunable resource scaling.

We focus on the Select-Swap (SS) model as it allows for a flexible exchange between T-depth and T-count via a parameter $\lambda \in \{0, \ldots, \log N\}$. For a dataset of size $N = 2^n$:

- **Minimal T-Count Configuration ($\lambda \approx 0$):** This configuration minimizes the total number of physical resources. It achieves a T-gate count of $\mathcal{O}(N)$ and uses $\mathcal{O}(N)$ ancilla qubits. However, the circuit depth scales linearly as $\mathcal{O}(N)$, which may mitigate quantum speedups in time-critical applications.
- **Minimal T-Depth Configuration ($\lambda \approx n$):** To preserve the exponential or polynomial speedup of quantum algorithms, one typically prioritizes circuit depth. In this regime, the SS model achieves a T-depth of $\mathcal{O}(\text{polylog}(N))$ (specifically $\mathcal{O}(n)$). The trade-off is a significant increase in spatial overhead, requiring $\mathcal{O}(N^2)$ T-count and ancilla qubits.

In our resource estimation, we assume the availability of QRAM optimized for T-depth (the second configuration) to ensure the overall algorithmic time complexity remains logarithmic with respect to the input dimension N. While this implies a hardware cost scaling polynomially with N, the query depth remains $\mathcal{O}(\text{polylog}(N))$, consistent with the requirements for maintaining the asymptotic quantum advantage claimed in our framework.

B Implementation details

B.1 DCD Protocol

The implementation of DCD protocol to measure a quantum state is based upon the Quantum Discrete Chebyshev Transformation (QDCT), the function of which can be written as

$$U_{DCT}|i\rangle = |T_i\rangle = \sum_j T_i(x_j)|j\rangle. \tag{9}$$

QDCT can be realized with elementary gates and the Quantum Fourier Transformation circuits as shown in Klappenecker & Rotteler (2001), with quantum overhead scaling logarithmic with the system size. The DCD protocol is to obtain the coefficients of each Chebyshev basis by quantum amplitude estimation, the complexity of which comes up naturally now:

proof of Theorem 3.1. The coefficients estimation stage includes 3 parts: state preparation, QDCT, and amplitude estimation. The complexity of state preparation and QDCT is $O(\text{polylog } d \times C_\psi)$. The amplitude estimation will multiply the complexity by a factor $O(\frac{1}{\delta})$. There are r coefficients required to be estimated. Therefore, the overall complexity is $O(r \cdot C_\psi/\delta)$. The coefficients loading and state computation compose the state re-preparation stage. The cost of them is linear with r, which gives the claimed re-preparation overhead. □

B.2 Residual Layer

To construct deep quantum neural networks, we introduce a quantum analogue of the classical residual block, inspired by ResNet architectures. This block enables the training of deeper models by using shortcut connections to mitigate vanishing gradient problems. A single block operates on a quantum state encoding a feature map and is composed of a main path and a shortcut path. The data flow within the block is managed by a Data Transfer Module (DTM), which handles state preparation from classical data via QRAM and measurement for intermediate classical processing.

A typical quantum residual block executes the following sequence:

1.**Main Path:** The input state $|\psi_{\text{in}}\rangle$, encoding the feature map X, is processed sequentially by a quantum convolutional layer (U_{qConv}), a quantum batch normalization layer (U_{qBN}), and a quantum ReLU activation (U_{QReLU}). This sequence may be repeated, as in standard ResNet blocks.

2.**Shortcut Path:** The original input state $|\psi_{\text{in}}\rangle$ is preserved.

3.**Addition & Final Activation:** The output state from the main path, $|\psi_{\text{main}}\rangle$, is added to the shortcut state $|\psi_{\text{in}}\rangle$ using a quantum arithmetic adder from the QAM. A final ReLU activation, U_{QReLU}, is applied to the resulting state to produce the block's output state, $|\psi_{\text{out}}\rangle$.

Quantum Convolutional Layer (qConv). The qConv layer performs convolution using the Quantum Arithmetic Module (QAM). Its goal is to transform an input feature map state $|\psi_X\rangle$ into an output state $|\psi_Y\rangle$ where Y is the convolution of X with a classically-defined kernel K satisfying

$$Y_{ijc} = (\text{bias})_c + \sum_{c',\Delta i,\Delta j} K_{c,c',\Delta i,\Delta j} \cdot X_{i+\Delta i,j+\Delta j,c'}. \tag{10}$$

The operation can be described as follows: for each output pixel position (i, j, c), the QAM applies a unitary U_{qConv} that computes the dot product arithmetically:

$$U_{\text{qConv}} : |i,j,c\rangle\,|0\rangle \to |i,j,c\rangle\,|Y_{ijc}\rangle\,. \tag{11}$$

This computation leverages the quantum adders and multipliers within the QAM to perform the operation in superposition across all output positions. The complexity comes as follows:

Lemma B.1 (Convolutional Layer)**.** *Given the shape of the input tensor X be (B, C, H, W) together with the kernel shape (C, C, K, K), the gate count of U_{qConv} is $O(CK^2)$, while the gate depth is $O(\log(CK))$.*

Proof. In the proof, we disregard the number of bits for the data, as its generalization to more bits of floating-point numbers is straightforward. First, we use the quantum input model for classical data loading with data replication to prepare

$$|i,j,c\rangle|0^{\otimes c+2k}\rangle \to |i,j,c\rangle \bigotimes_{\Delta i,\Delta j,c'} |X_{i+\Delta i,j+\Delta j,c'}\rangle, \tag{12}$$

where $c = \lceil \log_2 C \rceil, k = \lceil \log_2 K \rceil$.The data replication process only increases the word size for data loading, which contributes polylogarithmically to the complexity. The data loading is similar for

$$|c\rangle\,|0\rangle \to |c\rangle \bigotimes_{\Delta i,\Delta j,c'} |K_{\Delta i,\Delta j,c,c'}\rangle. \tag{13}$$

Set U_{qConv} to be the inner product circuit on the last two registers, and the complexity is then $O(CK^2)$, as discussed in Section A.2. We achieve the claimed operation and the claimed complexity. The gate depth can be further optimized to $O(\log(CK))$ by applying quantum adders and multipliers simultaneously. □

Quantum Batch Normalization Layer (qBatchNorm). The qBatchNorm layer, crucial for stabilizing training, is implemented in a hybrid quantum-classical manner. Due to the difficulty of computing global statistics (mean and variance) on a quantum state directly, we first perform a measurement on the state produced by the qConv layer. This DTM operation yields a classical snapshot of the feature map data. From this classical data, we compute the batch mean μ and variance σ^2. These classical parameters are then used to configure a quantum arithmetic circuit U_{qBN} within the QAM. This circuit applies the normalization transformation element-wise in superposition:

$$U_{\text{qBN}} : |y\rangle \to |\gamma \frac{y-\mu}{\sqrt{\sigma^2+\delta}} + \beta\rangle\,, \tag{14}$$

where γ and β are learnable classical parameters, and $|y\rangle$ is a register digitally encoding a single feature value. The complexity of the statistics estimation, which corresponds to the batch size B', is $O(B'CK^2)$. The normalization, involving only single-qubit operations, costs $O(1)$.

Quantum ReLU. The subsequent Quantum ReLU (U_{QReLU}) is similarly implemented as an arithmetic comparison circuit within the QAM, applying $|y\rangle \to |\max(0, y)\rangle$.

Combining the discussion above naturally gives the overall complexity of qResNet.

Proof of Theorem 3.2. The convolution operation is the main bottleneck, which costs $\tilde{O}(CK^2)$ by Lemma B.1. Considering the sampling overhead, the overall gate complexity is then $\tilde{O}(CK^2 \times S(B, C, H, W))$. Furthermore, the input models are respectively queried for X and K twice in each implementation, since one is required for uncomputing. □

Algorithm 2: Quantum Residual Block

```
Require: Input quantum state $|\psi_X\rangle$; Classical kernel $K$; Parameters $\gamma, \beta$.
Ensure : Output quantum state $|\psi_{\text{out}}\rangle$.
/* Main Path                                                    */
$|\psi_{\text{conv}}\rangle \leftarrow U_{\text{qConv}}(K)\,|\psi_X\rangle$;
/* Hybrid Batch Normalization Step                              */
$X_{\text{conv}} \leftarrow \text{Measure}(|\psi_{\text{conv}}\rangle)$;
$\mu, \sigma^2 \leftarrow \text{ComputeBatchStats}(X_{\text{conv}})$;
$|\psi_{\text{BN}}\rangle \leftarrow U_{\text{qBN}}(\mu, \sigma^2, \gamma, \beta)\,|\psi_{\text{conv}}\rangle$;
$|\psi_{\text{main}}\rangle \leftarrow U_{\text{QReLU}}\,|\psi_{\text{BN}}\rangle$;
/* Shortcut and Addition                                        */
$|\psi_{\text{shortcut}}\rangle \leftarrow |\psi_X\rangle$;
$|\psi_{\text{sum}}\rangle \leftarrow \text{QuantumAdd}(|\psi_{\text{main}}\rangle, |\psi_{\text{shortcut}}\rangle)$;
/* Final Activation                                             */
$|\psi_{\text{out}}\rangle \leftarrow U_{\text{QReLU}}\,|\psi_{\text{sum}}\rangle$;
return $|\psi_{out}\rangle$;
```

B.3 Quantum Transformer

This section details the quantum implementation of the Transformer architecture, which, like the quantum ResNet, is constructed from modular components. It leverages the Quantum Arithmetic Module (QAM) and the Quantum Linear Algebra Module (QLA). Without loss of generality, our analysis concentrates on a single batch and one-bit data. We assume that the input tensor of each building block $X^{(in)} \in \mathbb{R}^{N \times d}$ is quantum digitally encoded by the operator O_X, which is a quantum digital encoding of the input tensor $X^{(in)}$:

$$O_X |i\rangle |0^{\otimes d}\rangle = |i\rangle \bigotimes_j |X^{(in)}_{i,j}\rangle . \tag{15}$$

where N is the sequence length and d is the embedding dimension. For long-context tasks where $N \gg d$, the key to efficiency lies in how these modules handle the different dimensions: the large dimension N is parallelized over using index registers, while the smaller dimension d is processed arithmetically.

B.3.1 Quantum multi-head self-attention mechanism

At the core of the quantum encoder, the self-attention mechanism is a hybrid of QAM-based arithmetic for local operations and QLA-based matrix multiplication for the final aggregation. The parallelism over the sequence length N is achieved by encoding the token indices into dedicated quantum registers, allowing the QAM to operate on all elements in superposition.

Q, K, V Projection and Score Calculation. The initial step projects the input state $|X^{(in)}\rangle$ into Query ($\boldsymbol{Q}$), Key ($\boldsymbol{K}$), and Value ($\boldsymbol{V}$) representations. This is N independent multiplications on the d-dimensional vectors. This is performed by the QAM, conditioned on an index register $|i\rangle$ spanning the N tokens. Considering the h heads, the output quantum circuit O_Q (similarly for K, V) should be

$$O_Q |i_N, i_h\rangle |0\rangle = |i_N, i_h\rangle \bigotimes_{0 \le j < d_k} |Q_{i_h, i_N, j}\rangle , \tag{16}$$

where i_N, i_h, j are respectively the sequence, head, and dimension indices. Given the query weight matrix encoded by $O_{W,q}$:

$$O_{W,q} |i_h\rangle |0\rangle = |i_h\rangle \bigotimes_{\substack{0 \leq j < d_k, \\ 0 \leq k \leq d}} |(W_Q)_{j+i_h h,k}\rangle \equiv |i_h\rangle |W_{Q,i_h}\rangle , \tag{17}$$

where $d_k = d/h$, O_Q can be realized as follows:

$$\begin{aligned}
&|i_N, i_h\rangle |0\rangle |0\rangle |0^{\otimes d_k}\rangle \\
&\xrightarrow{\text{Data Loading}} |i_N, i_h\rangle \bigotimes_{0 \leq j < d} |X^{(in)}_{i_N,j}\rangle \bigotimes_{\substack{0 \leq k < d_k, \\ 0 \leq l \leq d}} |(W_Q)_{kl}\rangle |0^{\otimes d_k}\rangle \\
&\xrightarrow{\text{Element wise multiplication}} |i_N, i_h\rangle |X^{(in)}_{i_N}\rangle |W_{Q,i_h}\rangle \bigotimes_{\substack{0 \leq \alpha < d_k, \\ 0 \leq \beta \leq d}} |(W_Q)_{\alpha+i_h h,\beta} X^{(in)}_{i_N,\beta}\rangle |0^{\otimes d_k}\rangle \\
&\xrightarrow{s \text{ steps addition}} |i_N, i_h\rangle |X^{(in)}_{i_N}\rangle |W_{Q,i_h}\rangle |W \circ X\rangle \bigotimes_{0 \leq \alpha < d_k} |\sum_{0 \leq \beta < d} (W_Q)_{\alpha+i_h h,\beta} X^{(in)}_{i_N,\beta}\rangle \\
&\xrightarrow{\text{Rewriting \& Uncomputing}} |i_N, i_h\rangle \bigotimes_{0 \leq \alpha < d_k} |Q_{i_N,\alpha+i_h h}\rangle \equiv |i_N, i_h\rangle |Q_{i_h,i_N}\rangle ,
\end{aligned} \tag{18}$$

where $|W \circ X\rangle$ are simplified notations of the states generated in the second step. The derivation is almost the same for K, V.

The subsequent calculation of the attention scores, $\boldsymbol{S} = \boldsymbol{QK}^T/\sqrt{d_k}$, which results in an $N \times N$ matrix, follows a similar procedure. To compute all N^2 scores in parallel, we use two index registers, $|i\rangle$ and $|j\rangle$. An arithmetic circuit within the QAM then executes the dot product conditioned on these indices. The transformation on the quantum state can be abstractly represented as:

$$U_{\text{dot-prod}} : |i, j, i_h\rangle |Q_{i,i_h}\rangle |K_{j,i_h}\rangle |0\rangle \to |i, j\rangle |Q_{i,i_h}\rangle |K_{j,i_h}\rangle |S_{ij,i_h}\rangle . \tag{19}$$

Here, the state $|i, j\rangle$ acts as a control, specifying which dot product to compute, while the operation itself happens on the data registers. The states $|Q_i\rangle$ and $|K_j\rangle$ represent the necessary data for the computation, which come from the previous discussion. This explicitly shows how the large $N \times N$ dimensional workload is handled through quantum parallelism rather than matrix size.

The cost of the preparation of K, Q is simply twice the single cost, which is $O(d_k d) = O(d^2)$, equal to the total number of components involved. The circuit of the attention scores computation implements the inner product of dimension d_k, whose complexity is $O(d_k)$. They constitutes the overall $O(d^2)$ complexity.

Softmax A full quantum implementation of the softmax function is notoriously difficult. We therefore adopt a hybrid quantum-classical approach. The state encoding the unnormalized score matrix $\boldsymbol{S}$ (as constructed in Eq. 19) is measured using the DTM. The $N \times N$ matrix of scores is then post-processed classically to compute the final attention matrix $\boldsymbol{A} = \text{softmax}(\boldsymbol{S})$. After uncomputing, we have built the arithmetic circuit $U_{A,arith}$:

$$U_{A,arith} |i, j, i_h\rangle |0\rangle = |i, j, i_h\rangle |A_{i,j,i_h}\rangle . \tag{20}$$

Weighted Sum The next step is the product $\boldsymbol{AV}$. Here, $\boldsymbol{A}$ is a large, dense $N \times N$ matrix. Given our assumption that $N \gg d$, this large matrix multiplication is precisely the task for which the QLA is designed. The classical matrix $\boldsymbol{A}$ is used to construct its block-encoding unitary, U_A, where we have

$$\langle 0| U_A |0\rangle = \frac{1}{N} \sum_{i_h} |i_h\rangle \langle i_h| \otimes A_{i_h}. \tag{21}$$

Here $A_{i_h} = \text{softmax}(Q_{i_h} K_{i_h}^T/\sqrt{d_k})$. This U_A is built by the basic dense block-encoding Gilyén et al. (2019) based on the arithmetic circuit $U_{A,arith}$:

$$(H \otimes I \otimes I)(\text{SWAP} \otimes I) U^\dagger_{A,arith} U_{dac} U_{A,arith} (H \otimes I \otimes I), \tag{22}$$

where U_{dac} is the quantum circuit transforming bit strings into amplitude as introduced in Mitarai et al. (2019). The QLA then efficiently applies this unitary to the quantum state encoding the $\boldsymbol{V}$ matrix, which comes similarly from the arithmetic circuit and U_{dac}. The gate complexity comes mainly from several queries to $U_{A,arith}$, and therefore remains invariant.

Multi-Head Parallelism. After the weighted sum step, we obtain the quantum circuit realizing

$$U_{ws} \left| i_h, i \right\rangle \left| 0 \right\rangle = \left| i_h, i \right\rangle \frac{1}{N} \sum_j (A_{i_h} V_{i_h})_{ij}. \tag{23}$$

Note that when the input state is

$$|\vec{1}\rangle \frac{1}{\sqrt{d_k}} \sum_{i_h} \left| i_h \right\rangle, \tag{24}$$

the multi-head mechanism is naturally realized, with the output being

$$U_{ws} |\vec{1}\rangle \left| i \right\rangle \left| 0 \right\rangle = \frac{1}{N\sqrt{d_k}} \left| i \right\rangle \sum_{i_h, j} (AV)_{i_h, ij}. \tag{25}$$

The final linear projection $\boldsymbol{W}_O$ can be processed as an operator applying to the amplitude using similar method as above.

B.3.2 Quantum Feed-Forward Network (FFN)

Each Transformer block contains a position-wise Feed-Forward Network (FFN), applied independently to each of the N token positions. This sub-layer is implemented entirely using the QAM. The mechanism is identical to that in the attention layer: the FFN's arithmetic circuits (two linear maps and a QReLU) are conditioned on an index register $\left| i \right\rangle$ that spans all N token positions, thus processing all tokens in parallel. A complete quantum Transformer block is then formed by enclosing both the multi-head attention and FFN modules within residual connections and layer normalization, which are also implemented as QAM-based arithmetic operations conditioned on the token index.

Proof of Theorem 3.3. An encoder layer contains two residual connection layers, an MHSA layer, and an FFN layer. The MHSA layer is the resource bottleneck. For the MHSA layer, the arithmetic part scales as $\tilde{O}(d^2)$. The overall gate complexity is then $\tilde{O}(d^2 \times S(B, N, d))$ after considering the sampling overhead. It requires 6 queries to X, where $U_{A,arith}$ and its conjugate costs 2, and the vector encoding of V costs 2. □

Algorithm 3: Quantum Transformer Block

```
Require: Input quantum state $|\psi_X\rangle$ encoding the sequence $X$; Classical parameters $\theta_{\text{Attn}}, \theta_{\text{FFN}}$
         for all layers.
Ensure : Output quantum state $|\psi_{\text{out}}\rangle$ after one Transformer block.
$|\psi_{\text{attn}}\rangle \leftarrow$ QuantumMultiHeadAttention($|\psi_X\rangle, \theta_{\text{Attn}}$);
/* Multi-Head Attention Sub-layer                                  */
/* Hybrid approach:  QAM for projections/scores, QLA for AV
   product.                                                        */
$|\psi_{\text{add1}}\rangle \leftarrow$ QuantumAdd($|\psi_X\rangle, |\psi_{\text{attn}}\rangle$);
/* Residual connection:  Position-wise Add via QAM.                */
$|\psi_{\text{norm1}}\rangle \leftarrow U_{\text{LayerNorm}}(|\psi_{\text{add1}}\rangle)$;
$|\psi_{\text{ffn}}\rangle \leftarrow U_{\text{FFN}}(|\psi_{\text{norm1}}\rangle, \theta_{\text{FFN}})$;
/* Feed-Forward Sub-layer                                          */
$|\psi_{\text{add2}}\rangle \leftarrow$ QuantumAdd($|\psi_{\text{norm1}}\rangle, |\psi_{\text{ffn}}\rangle$);
/* Residual connection:  Position-wise Add via QAM.                */
$|\psi_{\text{out}}\rangle \leftarrow U_{\text{LayerNorm}}(|\psi_{\text{add2}}\rangle)$;
return $|\psi_{out}\rangle$;
```

C Quantum-Accelerated Backpropagation

Training deep neural networks relies on backpropagation, which systematically computes the gradient of the loss function with respect to the model's weights. We propose a quantum-accelerated approach for this process, where the core matrix operations of the chain rule are mapped to our QAM and QLA modules. The overall process remains hybrid: gradients are typically stored and updated classically, but their computationally expensive calculation is offloaded to the quantum processor.

To illustrate the principle, we consider the backward pass through a single linear layer, defined by the forward pass $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{X}$. Here, $\boldsymbol{W}$ is a $d \times d$ weight matrix and $\boldsymbol{X}$ is a $d \times N$ matrix representing N data points. Given the gradient from the subsequent layer, $\partial L/\partial \boldsymbol{Y}$ (a $d \times N$ matrix), we must compute two quantities: the gradient to be propagated backward, $\partial L/\partial \boldsymbol{X}$, and the gradient for updating the weights, $\partial L/\partial \boldsymbol{W}$.

Gradient Calculation for Weights ($\partial L/\partial \boldsymbol{W}$): Handled by QLA. The gradient with respect to the input is given by the chain rule:

$$\frac{\partial L}{\partial \boldsymbol{X}} = \boldsymbol{W}^T \frac{\partial L}{\partial \boldsymbol{Y}}. \tag{26}$$

This is a $(d \times d) \times (d \times N)$ matrix multiplication. Critically, this operation can be viewed as applying the small $(d \times d)$ matrix $\boldsymbol{W}^T$ to each of the N columns of the incoming gradient $\partial L/\partial \boldsymbol{Y}$. This is a "position-wise" operation, perfectly suited for the QAM. By conditioning on an index register $|j\rangle$ spanning the N columns, the QAM can perform all N matrix-vector products in parallel, arithmetically processing the d-dimensional vectors in superposition.

Gradient Calculation for Weights ($\partial L/\partial \boldsymbol{W}$): Handled by QLA. The gradient with respect to the weights is an outer product:

$$\frac{\partial L}{\partial \boldsymbol{W}} = \frac{\partial L}{\partial \boldsymbol{Y}} \boldsymbol{X}^T. \tag{27}$$

This is a $(d \times N) \times (N \times d)$ matrix multiplication, resulting in a $d \times d$ gradient matrix. Each element $(\partial L/\partial \boldsymbol{W})_{ij}$ is the inner product of the i-th row of $\partial L/\partial \boldsymbol{Y}$ and the j-th row of $\boldsymbol{X}$. Both are vectors of length N. Given our assumption that $N \gg d$, these are high-dimensional inner products. This task is ideal for the QLA's inner product estimation capability Xiong et al. (2024). Instead of performing a full matrix multiplication, the QLA can be configured to efficiently estimate the d^2 required inner products between the corresponding N-dimensional quantum states, yielding the elements of the weight gradient.

This strategic division of labor is fundamental to our training approach. The QAM handles the backward flow of gradients through the network's data path by parallelizing over the sequence/batch dimension N. The QLA, in turn, handles the most intensive gradient calculations for weights, which involve contractions over this large N dimension. This transforms the most demanding parts of backpropagation into potentially tractable quantum computations, paving the way for end-to-end quantum-accelerated training.