Hybrid Parameterized Quantum States for Variational Quantum Learning

Chen-Yu Liu
National Taiwan University

Abstract

Variational quantum learning faces practical challenges in the noisy intermediate-scale quantum (NISQ) era. Parameterized quantum circuit (PQC) models suffer from statistical uncertainty due to finite-shot measurements and are highly sensitive to quantum noise, while purely classical approximations like neural quantum states (NQS) lack access to genuine quantum correlations and are limited in scalability. This work introduces Hybrid Parameterized Quantum States (HPQS), a general-purpose modeling framework that interpolates between quantum and classical parameterizations. HPQS combines PQC-based measurements with neural estimators via a blending mechanism and postprocessing functions, enabling enhanced, shot-efficient evaluation under hardware constraints. We demonstrate HPQS across three representative quantum learning tasks: (1) Expectation-based QML, where HPQS yields higher classification accuracy than PQC-only and NQS-only baselines under limited quantum measurements. (2) Quantum-Train, where HPQS generates the entire parameter set of classical networks using polylogarithmic trainable variables; and (3) Quantum Parameter Adaptation (QPA), where HPQS produces LoRA adapter parameters for fine-tuning large language models like GPT-2 and Gemma-2 with improved perplexity under low-shot conditions; Together, these results position HPQS as a scalable, noise-resilient approach for variational quantum learning, compatible with both current NISQ hardware and future fault-tolerant architectures.

1 Introduction

Quantum machine learning (QML) is a rapidly growing field that seeks to leverage the representational power of quantum systems to improve learning performance in classification, generative modeling, and reinforcement learning tasks [1–5]. Among various approaches, *variational quantum algorithms (VQAs)* have emerged as a central paradigm due to their compatibility with near-term quantum hardware. Despite inherent limitations in circuit width and depth, they have demonstrated potential quantum advantages on carefully constructed problem instances [6–8]. These methods rely on *parameterized quantum circuits (PQCs)* to represent learnable quantum states, where model training is carried out by adjusting circuit parameters to minimize a loss function through repeated quantum measurements. However, the practical deployment of PQC-based models faces two critical limitations: (1) the inefficiency of finite-shot measurements, which introduces statistical noise and limits the accuracies of probability and expectation value estimation [8, 9], and (2) the scalability bottleneck of circuit size and depth, constrained by hardware fidelity and decoherence in today's noisy intermediate-scale quantum (NISQ) systems [10].

Recent efforts have proposed solutions such as classical shadow tomography for more efficient expectation estimation [11, 12], error mitigation methods to reduce noise bias [13, 14], and quantum error correction (QEC) techniques that protect logical qubits from noise [15, 16]. While these
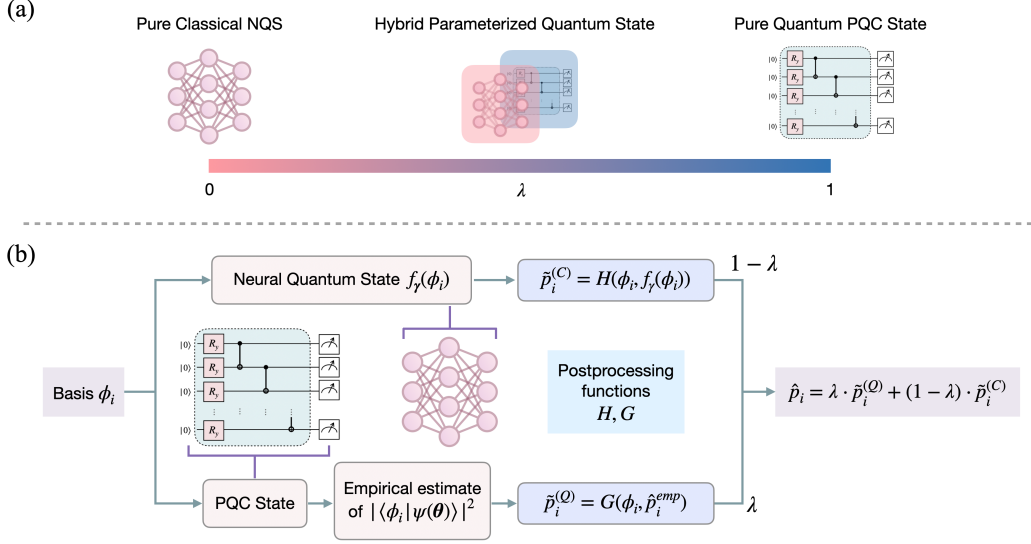
Figure 1: Overview of the HPQS framework. (a) HPQS interpolates between a pure classical NQS and a pure quantum PQC state using a blending coefficient $\lambda \in [0, 1]$. (b) Architecture of HPQS: Given a basis state ϕ_i, both a classical NQS $f_\gamma(\phi_i)$ and a PQC-based quantum state $|\psi(\boldsymbol{\theta})\rangle$ are evaluated. The classical and quantum outputs are then postprocessed by functions H and G, respectively, to produce $\tilde{p}_i^{(C)}$ and $\tilde{p}_i^{(Q)}$. The final probability estimate $\hat{p}_i$ is computed as a weighted combination of the two, enabling robust and shot-efficient variational quantum learning.

approaches show promise, they often introduce additional layers of computational or hardware complexity. For instance, QEC schemes typically require tens to hundreds of physical qubits to encode a single logical qubit, making them challenging to deploy in practical large-scale learning settings with current hardware.

While QML has predominantly focused on quantum circuits, in the classical computational perspective of quantum system simulation, *Neural Quantum States (NQS)*, which are classical neural network models that approximate quantum state distributions over computational basis states, has been proposed [17–19]. NQS-based models are powerful, expressive, and easy to optimize using backpropagation. They have shown promising results in simulating quantum systems and generative modeling [20, 21]. Given the nature of pure classical simulation, NQS lacks access to real quantum data and offers no inherent guarantee that the modeled distributions reflect physical quantum states unless specifically constrained [22–24].

From the perspective of PQC-based models, incorporating the inductive structure of NQS can alleviate the statistical inefficiency introduced by finite-shot quantum measurements. Classical neural networks can be trained to concurrently estimate measurement probabilities, providing a smooth, differentiable, and noise-free approximation of the quantum state. This not only potentially improves learning efficiency under limited quantum samples but also mitigates the impact of quantum hardware noise during training. Conversely, from the perspective of NQS-based models, introducing PQC components allows the network to leverage physically grounded quantum information that naturally captures entanglement, superposition, and other structural correlations inherent in quantum systems. The representation induced by PQC is known to be highly expressive with a relatively small number of parameters [25–27]. As we demonstrate in this work, the complementary interaction between classical and quantum components leads to empirical performance gains when either neural or quantum parameters are incorporated.

Nevertheless, in the low-qubit regime (dozens of qubits) that is possible to be approximated by NQS efficiently and under NISQ condition of quantum system, it becomes promising to **combine finite-measurements PQC with NQS**. This enables hybrid models that both learn from classical approximations and incorporate partial quantum state information. Such hybridization is particularly

compelling when only a subset of measurement outcomes is available or when the number of shots is limited.

In this work, we introduce *Hybrid Parameterized Quantum States (HPQS)*, a variational modeling framework that unifies the expressiveness of NQS with the physical grounding of PQC states. HPQS defines a learnable quantum-classical state that integrates empirical quantum measurements with classical neural estimators, enabling noise-resilient and shot-efficient variational learning under sparse, noisy, or limited-access quantum observations. Key contributions of this work are as follows:

- **Hybrid Formulation**: HPQS is formulated as a new class of variational quantum states that interpolates between PQC and NQS, enabling enhanced estimation of measurement probabilities using partial quantum measurements and a classical neural estimator.
- **Generality**: The HPQS framework generalizes both PQC-only and NQS-only models as special cases and can be further reduced to other hybrid quantum-classical frameworks under certain conditions (Appendix C), demonstrating that HPQS serves as a unifying and generalizable framework.
- **Demonstrated Effectiveness**:
  The effectiveness of HPQS is demonstrated across three representative QML scenarios: two recently proposed tasks related to Quantum-Train, and one standard benchmark task involving expectation-based quantum classification.
    - In **Quantum-Train (QT)**, HPQS generates the full parameter set of classical neural networks using only polylogarithmic trainable variables.
    - In **Expectation-Based QML**, HPQS enables more stable and enhanced expectation value estimation under low-shot measurement regimes.
    - In **Quantum Parameter Adaptation (QPA)**, HPQS surpasses pure NQS and PQC settings with finite measurement shots in generating LoRA adapter weights for large language model fine-tuning task.
- **Long-Term Viability**: HPQS is positioned as a practical yet forward-compatible architecture for quantum–classical co-design, suitable for both current NISQ-era systems and future fault-tolerant quantum hardware. Notably, even fault-tolerant devices are constrained by finite-shot measurements, and HPQS remains beneficial in such regimes by enhancing estimation efficiency.

2 Related works

2.1 Parameterized Quantum Circuit States

PQC states form the cornerstone of variational quantum learning [25–27]. In this framework, a quantum state is defined by a parameterized ansatz, typically composed of alternating layers of entangling gates (e.g., CNOT) and single-qubit rotations (e.g., R_Y, R_Z). For n qubits, a PQC defines a variational quantum state of the form:

$$|\psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|0\rangle^{\otimes n}, \tag{1}$$

where $U(\boldsymbol{\theta})$ is a trainable quantum circuit parameterized by a vector of real-valued parameters $\boldsymbol{\theta}$ and $|0\rangle^{\otimes n}$ is an n-qubit initialized state. The optimization objective typically minimizes a classical loss function $\mathcal{L}$ that depends on either measurement probabilities or expectation values of observables. PQC-based quantum states offer strong expressivity and can be designed to be hardware-efficient, particularly when adapted to the constraints of NISQ devices [28, 29]. However, their practical deployment faces two major bottlenecks: (1) the *finite-shot measurement problem*, which introduces statistical uncertainty in estimating measurement probabilities or expectation values; and (2) the presence of hardware-induced quantum noise, which further distorts the observed distributions.

In an exact classical simulation of a variational quantum state $|\psi(\boldsymbol{\theta})\rangle$, the probability of measuring a computational basis state $|\phi_i\rangle$ is given precisely by calculating $|\langle\phi_i|\psi(\boldsymbol{\theta})\rangle|^2$. However, on real quantum hardware, this probability must be estimated via repeated sampling (shots), which introduces statistical fluctuations. Let $\hat{P}(|\phi_i\rangle)$ denote the empirical estimate of this probability after n'_{shot} measurement repetitions. The expression $n'_{\text{shot}} = n_{\text{shot}}/2^n$ provides an approximate scale for the average number of shots per basis state, where n_{shot} is the total number of measurements and 2^n

is the Hilbert space dimension. This estimate is intended only as a guideline for scaling behavior, since measurement outcomes are not uniformly distributed and many basis states may receive few or no samples in practice. Hoeffding's inequality provides an upper bound on the probability that this estimate deviates from the true value by at least $\epsilon > 0$:

$$P\left(\left|\hat{P}(|\phi_i\rangle) - \mathbb{E}[P(|\phi_i\rangle)]\right| \geq \epsilon\right) \leq 2\exp(-2\epsilon^2 n'_{\text{shot}}). \tag{2}$$

This bound highlights a key limitation of PQC-based methods: accurately estimating the measurement probability of each basis state requires a large number of shots, particularly when the desired precision ϵ is small or the number of relevant basis states grows exponentially with the number of qubits (see Appendix D). This statistical bottleneck has motivated the development of shot-efficient quantum state estimation techniques, such as shadow tomography [12, 30, 31], which aim to infer properties of quantum states from a limited number of measurements. While these methods can reconstruct global information from fewer samples, their primary application has focused on estimating the expectation values of observables rather than recovering the full probability distribution over basis states.

Another major challenge is the presence of *quantum noise*, which arises from gate imperfections, decoherence, and readout errors in NISQ-era devices [10, 32]. As quantum circuits grow in depth or width, the accumulated noise compounds, leading to degraded output fidelity and unreliable measurements. This effect is particularly problematic in variational algorithms, where the learning signal (e.g., gradients or expectation values) may be obscured or biased by stochastic hardware noise. In practice, mitigating these errors often requires significant overhead, such as repetition for averaging, error correction codes, or post-processing techniques, all of which increase computational and physical resource demands [13–16].

Despite these limitations, PQC states remain a powerful tool in quantum machine learning. They naturally encode quantum correlations such as entanglement and superposition, which are difficult to capture using classical neural networks alone.

2.2 Neural Quantum States

NQS' are classical, parameterized models designed to approximate quantum probability distributions. Introduced in the context of variational Monte Carlo for quantum many-body systems [17–19, 22–24], NQS represent the wavefunction amplitudes or measurement probabilities (we use measurement probabilities in this work) using a classical neural network $f_\gamma(\phi_i)$, where ϕ_i denotes the basis state (typically expressed in binary vector) and γ are the learnable parameters of the network. In the most common formulation, the NQS defines a probability distribution over the computational basis as:

$$p_{\text{NQS}}(\phi_i) = \frac{f_\gamma(\phi_i)}{\sum_j f_\gamma(\phi_j)}, \tag{3}$$

where $f_\gamma(\cdot)$ is a positive-valued neural network (e.g., using softplus or sigmoid activations). The network is trained to approximate the target quantum distribution by minimizing a loss function derived from observed data.

NQS models offer several practical advantages. They are fully differentiable and can be optimized using standard gradient-based techniques such as stochastic gradient descent (SGD). Implemented on classical hardware, an NQS defines a function $f_\gamma : \{0,1\}^n \to \mathbb{R}_{\geq 0}$, where γ denotes the set of neural network parameters and n is the number of qubits (or bits in the basis state ϕ_i). Thanks to the universal approximation theorem [33, 34], a sufficiently expressive NQS (e.g., a multilayer perceptron with non-linear activations) can approximate any target distribution $p^*(\phi_i)$ to arbitrary accuracy on compact domains, i.e.,

$$|p_{\text{NQS}}(\phi_i) - p^*(\phi_i)| < \epsilon, \quad \forall \phi_i \in \{0,1\}^n, \text{ for any } \epsilon > 0. \tag{4}$$

This makes NQS a powerful surrogate for quantum states, particularly when access to real quantum measurement probabilities is limited, costly, or corrupted by finite-shot noise. However, NQS have no intrinsic access to quantum entanglement or nonlocal correlations unless such structure is explicitly learned from data or specified.

In this work, we reinterpret NQS as the classical backbone in our HPQS framework. The NQS serves as a flexible estimator that complements the partial and noisy output of PQCs. This neural component enables the model to infer unmeasured amplitudes or denoise quantum observations, thereby reducing the burden on quantum resources while retaining model expressivity.

3 Hybrid Parameterized Quantum States

3.1 General Formulation

To improve the robustness and expressivity of variational learning under finite-shot or noisy measurement conditions, we propose a hybrid framework called **Hybrid Parameterized Quantum States (HPQS)**. This framework blends quantum-generated information with classical neural estimators based on NQS.

Let $\hat{p}_i^{\text{emp}}$ denote the empirical estimate of $|\langle \phi_i | \psi(\boldsymbol{\theta}) \rangle|^2$ obtained from finite-shot quantum measurements of PQC. We define postprocessed quantum and classical predictions as:

$$\tilde{p}_i^{(Q)} = G(\phi_i, \hat{p}_i^{\text{emp}}), \quad \tilde{p}_i^{(C)} = H(\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)), \tag{5}$$

where $G : (\phi_i, \hat{p}_i^{\text{emp}}) \mapsto \mathbb{R}$ is a postprocessing function applied to quantum outputs (e.g., MLP decoder or tensor contraction), $H : (\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)) \mapsto \mathbb{R}$ is a postprocessing function applied to classical NQS outputs (e.g., alignment layer or task-adaptive transformation).

The final hybrid prediction for basis state ϕ_i is defined as:

$$\hat{p}_i = \lambda \cdot \tilde{p}_i^{(Q)} + (1 - \lambda) \cdot \tilde{p}_i^{(C)}, \tag{6}$$

with $\lambda \in [0, 1]$ as a scalar blending coefficient, which is considered as a hyperparameter. This unified formulation captures a wide range of hybrid strategies, such as:

- Direct blending when $G(x) = x$ and $H(x) = x$.
- Hybrid decoding pipelines when G and H are expressive decoders.
- When $\lambda = 1$, the model reduces to a pure quantum (PQC-based) formulation using only empirical measurement.
- When $\lambda = 0$, the model reduces to a purely classical (NQS-based) estimation framework.

With more possibilities investigated in Appendix C. By introducing G and H, HPQS enables the quantum and classical branches to be flexibly adapted to task-specific modalities or scales. It allows their outputs to be composed in a representation space before blending. A schematic drawing of HPQS is shown in Fig. 1.

From the perspective of the quantum circuit, this blended formulation allows PQC-derived outputs to be regularized and enhanced through classical estimation, mitigating finite-shot uncertainty and measurement noise. From the perspective of the classical neural estimator, HPQS introduces physically grounded quantum information into the learning process, enriching the model's expressivity with entanglement and non-classical correlations. Together, this hybrid mechanism enables scalable and noise-resilient variational learning, well suited for NISQ-era quantum systems, as demonstrated in the empirical experiment section.

Since HPQS includes an NQS component with universal approximation capacity, it inherits the ability to approximate any target probability distribution $p^*(\phi_i)$ over computational basis states. Specifically, for any $\epsilon > 0$, there exist parameters ϕ_{GH} and a blending weight $\lambda \in [0, 1]$ such that the hybrid estimate satisfies

$$|\hat{p}_i - p^*(\phi_i)| < \epsilon, \quad \forall \phi_i \in \{0, 1\}^n, \tag{7}$$

where $\hat{p}_i = \lambda \cdot \tilde{p}_i^{(Q)} + (1 - \lambda) \cdot \tilde{p}_i^{(C)}$ is the HPQS prediction, ϕ_{GH} are the corresponding parameters of $\hat{p}_i$ (constructing G, H, f_γ, and $\hat{p}_i^{\text{emp}}$). This theoretical expressivity complements the robustness brought by this parameterization design, especially under finite-shot or noisy conditions.

3.2 Parameter Generation via HPQS in Quantum-Train

In the QT framework [35–38], the goal is to generate the full set of trainable parameters $\boldsymbol{a} = (a_1, a_2, \ldots, a_m)$ of a classical neural network using a hybrid quantum-classical architecture. This replaces the need to train these parameters directly and instead formulates parameter generation as a variational learning task over a lower-dimensional latent space. Let m be the total number of

parameters in the target neural network. A PQC with $n = \lceil \log_2 m \rceil$ qubits and L layers is constructed to define the quantum state

$$|\psi(\boldsymbol{\theta})\rangle = \left(\prod_{i=1}^{n-1} \text{CNOT}^{i,i+1} \prod_{j=1}^{n} R_Y^j(\theta_j^{(L)}) \right)^L |0\rangle^{\otimes n}. \tag{8}$$

Here, each single-qubit rotation gate R_Y^j is parameterized by $\theta_j^{(L)}$, where j indexes the qubits, and L represents the circuit depth. The controlled-NOT (CNOT) gates create entanglement between qubits. After measurement, we obtain a finite-shot empirical estimate $\hat{p}_i^{\text{emp}}$ of the probability for each computational basis state $\phi_i \in \{0, 1\}^n$. At the same time, an NQS $f_{\boldsymbol{\gamma}}(\phi_i)$ is trained as a classical neural estimator for the outcomes of the same basis set. To map basis states and their associated quantum or classical probabilities to parameter values $a_i \in \boldsymbol{a}$, both branches are passed through postprocessing functions based on a tensor network architecture:

$$\tilde{a}_i^{(Q)} = G_{\text{TN}}(\phi_i, \hat{p}_i^{\text{emp}}), \quad \tilde{a}_i^{(C)} = H_{\text{TN}}(\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)). \tag{9}$$

G_{TN} and H_{TN} are tensor network-based mapping models [39], applied separately to the quantum and classical branches (see Appendix B for architectural details). Both functions can be viewed as structured decoders that learn to map basis-probability pairs to real-valued neural parameters.

The final HPQS-generated parameter is given by:

$$a_i = \lambda \cdot \tilde{a}_i^{(Q)} + (1 - \lambda) \cdot \tilde{a}_i^{(C)}, \tag{10}$$

with a scalar blending coefficient $\lambda \in [0, 1]$. This combination allows the model to leverage noisy but informative quantum measurements while compensating with a classical neural estimator where quantum information is limited. Once all a_i are generated via HPQS, they are used to instantiate the full set of weights in a classical neural network, which is then evaluated on a downstream task such as image classification. Only the quantum parameters $\boldsymbol{\theta}$, classical parameters ϕ, and tensor network weights in G_{TN} and H_{TN} are optimized. The number of trainable parameters is thus polylogarithmic in m [35], making this approach highly parameter-efficient. The use of tensor networks as the mapping model ensures expressive yet scalable transformation from basis information to model parameters. This design also allows seamless batching and parallelism during training, and unifies the interpretation of classical and quantum-generated states.

3.3 Expectation-Based Quantum Machine Learning with HPQS

In conventional QML, variational quantum circuits are often trained to minimize a loss function based on the expectation value of an observable [2]. For classification tasks, this typically involves encoding input data into a quantum state, applying a PQC, and measuring observables to obtain scalar values (e.g., logits or class scores). The predicted model output is then derived from these expectation values, following the approach in prior work [40].

We extend this approach through the HPQS framework by blending quantum-derived expectation values with classical predictions from a neural network model. In this setup, the quantum branch processes input x by preparing a parameterized quantum state $|\psi(\boldsymbol{\theta}, x)\rangle$ with angle encoding method, followed by measurement of a Pauli observable $\hat{Z}$ to yield the empirical expectation value:

$$\hat{y}_{\text{quantum}} = \hat{\mathbb{E}}[\hat{Z}] = \sum_i z_i \hat{p}_i^{\text{emp}}, \quad \hat{p}_i^{\text{emp}} = \text{empirical estimate of } |\langle \phi_i | \psi(\boldsymbol{\theta}, x)\rangle|^2. \tag{11}$$

This scalar output may be noisy or imprecise under finite-shot conditions, especially for deep circuits or large input spaces. Simultaneously, a classical neural network $f_{\boldsymbol{\gamma}}(x)$ is trained to produce a prediction from the same input x. Rather than choosing one branch exclusively, HPQS blends the two via postprocessing functions:

$$\tilde{y}^{(Q)} = G(\hat{y}_{\text{quantum}}), \quad \tilde{y}^{(C)} = H(f_{\boldsymbol{\gamma}}(x)), \tag{12}$$

where G is a postprocessing function applied to the quantum expectation value (e.g., scaling or nonlinearity), H is a transformation of the classical output (e.g., projection, batch normalization, or logit shaping). The actual H and G used in this example are described in the Appendix B.

The final HPQS prediction is given by:

$$\hat{y} = \lambda \cdot \tilde{y}^{(Q)} + (1 - \lambda) \cdot \tilde{y}^{(C)}, \tag{13}$$

where $\lambda \in [0, 1]$ is a blending coefficient. This structure allows HPQS to operate as a hybrid inference model, using quantum expectation values when reliable and falling back on classical learning when quantum resources are sparse or noisy. The output $\hat{y}$ is used as input to a loss function (e.g., cross-entropy for classification or mean squared error for regression), and gradients are propagated through both the quantum and classical branches. This hybrid formulation unifies PQC-based learning with classical machine learning in a scalable and shot-efficient manner.

This configuration is particularly well-suited for low-qubit, expectation-based quantum learning tasks such as binary classification. As demonstrated in Section 4, HPQS achieves more stable and improved learning performance compared to PQC-only baselines under finite-shot conditions, while remaining competitive with purely classical methods. Notably, HPQS mitigates the shot inefficiency commonly observed in PQC by directly processing task inputs, reflecting its core principle of seamlessly blending quantum measurements with classical inference.

3.4 Enhancing LLM Fine-Tuning with HPQS-based Quantum Parameter Adaptation

Large language models (LLMs) such as GPT-2 and Gemma-2 [41, 42] have demonstrated strong generalization across a wide range of natural language tasks. However, fine-tuning such models remains computationally expensive due to their large number of trainable parameters. To address this, parameter-efficient fine-tuning (PEFT) methods, such as Low-Rank Adaptation (LoRA) [43], adapt only a small subset of parameters while freezing the rest of the model. Quantum Parameter Adaptation (QPA) [44] extends the QT approach to the PEFT setting, to generate only the adaptation-specific parameters. Unlike full model generation, QPA focuses on fine-tuning modules such as LoRA matrices, which are significantly smaller in dimension but crucial for downstream task adaptation. In QPA, the LoRA update to a pretrained weight matrix $W_0 \in \mathbb{R}^{d \times k}$ is expressed as:

$$W_0 + \Delta W = W_0 + BA, \tag{14}$$

where $A \in \mathbb{R}^{r \times k}$, $B \in \mathbb{R}^{d \times r}$, and $r \ll \min(d, k)$ defines the rank of the low-rank update. The goal of QPA is to generate the entries of A and B via a hybrid quantum-classical procedure, while minimizing the number of trainable variables in the quantum circuit and classical neural estimator. To scale QPA to large models, we adopt a batched parameter generation strategy [45] as in the original QPA proposal. The full adaptation vector $\boldsymbol{a} \in \mathbb{R}^{r(d+k)}$ is partitioned into n_{ch} chunks of size n_{mlp}, with each chunk generated independently using HPQS. For each chunk index i, a basis ϕ_i for a quantum state $|\psi(\boldsymbol{\theta})\rangle$ is prepared and measured to yield $\hat{p}_i^{\text{emp}}$, while an NQS estimator provides the corresponding classical value $f_{\boldsymbol{\gamma}}(\phi_i)$. These values are then passed through tensor network decoders G and H as in the QT setup:

$$\tilde{a}_i^{(Q)} = G_{\text{TN}}(\phi_i, \hat{p}_i^{\text{emp}}), \quad \tilde{a}_i^{(C)} = H_{\text{TN}}(\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)), \tag{15}$$

and the HPQS estimate of the adaptation parameters becomes:

$$\boldsymbol{a} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{n_{\text{ch}}}), \tag{16}$$

$$\hat{a}_i = \lambda \cdot \tilde{a}_i^{(Q)} + (1 - \lambda) \cdot \tilde{a}_i^{(C)}, \quad \forall i \in \{1, 2, \ldots, n_{\text{ch}}\} \tag{17}$$

$$\hat{a}_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,j}), \quad \forall j \in \{1, 2, \ldots, n_{\text{mlp}}\}. \tag{18}$$

This process is repeated across all n_{ch} chunks to produce the complete set of LoRA weights. Importantly, this approach reduces the number of required qubits from $N = \lceil \log_2 m \rceil$ to

$$N = \lceil \log_2 n_{\text{ch}} \rceil = \lceil \log_2 \left(\lceil \frac{m}{n_{\text{mlp}}} \rceil \right) \rceil, \tag{19}$$

allowing adaptation to large-scale models even under tight quantum hardware constraints. Additionally, the shot efficiency and noise robustness of HPQS enable stable fine-tuning with limited quantum samples, addressing a key limitation of the original QPA formulation, as noted in Appendix G of that work.

4 Empirical Experiments

4.1 Overall Performance

To validate the effectiveness of the proposed HPQS framework, we conduct experiments across three representative quantum machine learning tasks [1]: parameter generation (QT), expectation-based classification (QML), and parameter-efficient adaptation (QPA). For each task, we compare four configurations:

- **PQC (exact)**: full access to the quantum state vector via exact simulation.
- **PQC (finite)**: probabilities estimated from a finite number of measurement shots simulation.
- **NQS**: pure classical neural quantum state.
- **HPQS (finite)**: hybrid design combining NQS and PQC (finite).

Each experiment is conducted under both noise-free and IBM quantum noise models (see Appendix A), using a fixed shot budget for PQC-based configurations (e.g., 10×2^n, where 2^n corresponds to the Hilbert space size, denoted as HSS). All simulations and optimizations are implemented using PyTorch and TorchQuantum [40, 46].

Parameter Generation via HPQS in Quantum-Train. We evaluate HPQS in the context of full parameter generation for classical neural networks. The quantum output is postprocessed through a tensor-network-based mapping model and blended with its NQS counterpart. The task involves classifying the MNIST dataset [47] (10 classes) using a compact convolutional neural network (CNN) with 6690 target parameters, which are generated from a significantly smaller set of trainable variables. As shown in Table 1, HPQS achieves accuracy comparable to PQC (exact), while significantly outperforming PQC (finite) and approaching or exceeding the performance of NQS under limited-shot conditions. Each reported accuracy is averaged over three random seeds, and the detailed architecture and training settings are provided in Appendix B.

Table 1: Comparison of QT models on MNIST-10 classification under various access and shot regimes.

QT (MNIST-10)			
Model	**Testing Accuracy (%)**	**# Training Param.**	**Shot Count**
PQC (exact)	**90.59 ± 1.12**	904	∞
HPQS (finite)	86.28 ± 0.58	649	$10 \times$ HSS
HPQS (finite)	**87.64 ± 0.47**†	792	$10 \times$ HSS
NQS	76.13 ± 7.25	512	—
NQS	84.80 ± 5.45	2624	—
PQC (finite)	57.54 ± 5.61	1164	$10 \times$ HSS
PQC (finite)	65.83 ± 2.56	1424	$10 \times$ HSS

Expectation-Based Quantum Machine Learning with HPQS. We evaluate HPQS on a binary classification task involving MNIST digits 3 versus 6, following the setup described in [40]. In this setting, a variational quantum classifier generates scalar outputs based on measured expectation values. HPQS blends these quantum-derived values with classical predictions from a neural estimator. As shown in Table 2, HPQS outperforms both NQS and PQC under finite-shot conditions, achieving a peak accuracy of $\mathbf{90.66 \pm 1.09}\%$ using 20×HSS shots. Notably, it surpasses the performance of PQC (exact), which achieves $86.66 \pm 2.17\%$, and significantly outperforms PQC (finite), which degrades substantially under limited-shot settings (e.g., $46.21 \pm 9.26\%$ with $5\times$ HSS). Each result is averaged over three random seeds. These findings highlight HPQS's ability to enhance robustness against shot noise and deliver superior performance compared to either component individually under practical quantum constraints.

Enhancing LLM Fine-Tuning with HPQS-based Quantum Parameter Adaptation. To demonstrate the scalability of HPQS, we apply it to QPA for LoRA-based fine-tuning of large language models, including GPT-2 (80M) and Gemma-2 (2B), on the WikiText-2 dataset [48]. The low-rank

---

[1] The code is attached to the supplemental material.

matrices used for adaptation are generated via a batched HPQS parameter generation scheme. Fine-tuning is applied only to the final linear projection layer, with the remainder of the model kept frozen, as in the original QPA paper [44].

As shown in Table 3, HPQS achieves strong performance under limited quantum resources. On GPT-2, HPQS (finite) obtains a perplexity of 4.612 using only $1\times$ HSS shots, compared to 1.616 for PQC (exact), 6.975 for PQC (finite), and a much worse 153.629 for NQS. Similarly, on Gemma-2, HPQS (finite) achieves 1.467, closely matching PQC (exact) at 1.427, while outperforming both NQS (1.471) and PQC (finite) (1.472). Each reported number reflects the best result over three random seeds. These results confirm that HPQS can maintain competitive or superior performance compared to pure quantum or classical baselines, even when constrained to finite-shot quantum access. This demonstrates HPQS's effectiveness as a practical, scalable approach for low-resource fine-tuning in modern LLMs.

4.2 Robustness to Quantum Noise

We further evaluate the noise resilience of HPQS by simulating real IBM quantum noise models. Across all tasks, HPQS consistently outperforms PQC (finite) under noisy conditions, with relative degradation that is notably less severe than pure PQC models. This confirms HPQS's robustness in practical, imperfect quantum environments. Detailed comparisons are provided in Appendix A.

Table 2: Comparison of QML models under different access and shot regimes.

QML (MNIST-2)			
Model	**Testing Accuracy (%)**	**# Training Param.**	**Shot Count**
PQC (exact)	86.66 ± 2.17	40	∞
HPQS (finite)	88.97 ± 5.08	65	$5 \times$ HSS
HPQS (finite)	$\mathbf{90.66 \pm 1.09}$	65	$20 \times$ HSS
NQS	$\mathbf{89.67 \pm 5.30}^{\dagger}$	21	—
PQC (finite)	46.21 ± 9.26	40	$5 \times$ HSS
PQC (finite)	53.33 ± 6.61	40	$20 \times$ HSS

Table 3: Comparison of QPA models under different access and shot regimes.

Task	Model	Testing PPL	# Training Param.	Shot Count
QPA (GPT-2 & Wikitext-2)	PQC (exact)	**1.616**	22368	∞
QPA (GPT-2 & Wikitext-2)	HPQS (finite)	$\mathbf{4.612}^{\dagger}$	23585	$1 \times$ HSS
QPA (GPT-2 & Wikitext-2)	NQS	153.629	18113	—
QPA (GPT-2 & Wikitext-2)	PQC (finite)	6.975	22368	$1 \times$ HSS
QPA (Gemma-2 & Wikitext-2)	PQC (exact)	**1.427**	140432	∞
QPA (Gemma-2 & Wikitext-2)	HPQS (finite)	$\mathbf{1.467}^{\dagger}$	141585	$1 \times$ HSS
QPA (Gemma-2 & Wikitext-2)	NQS	1.471	136321	—
QPA (Gemma-2 & Wikitext-2)	PQC (finite)	1.472	140432	$1 \times$ HSS

5 Discussion and Conclusion

In this work, we introduced *Hybrid Parameterized Quantum States (HPQS)*, a unified framework that interpolates between PQC and NQS for variational quantum learning. By blending quantum measurement outcomes with classical neural estimators, HPQS enables scalable, shot-efficient, and noise-resilient learning suitable for near-term quantum devices. We validate HPQS across three representative tasks spanning increasing model complexity: expectation-based classification in QML, full parameter generation in QT, and fine-tuning of large language models in QPA. In all cases, we empirically demonstrate that HPQS outperforms PQC under finite-shot conditions and achieves comparable or better performance than PQC (exact) and NQS, while maintaining strong expressivity and robustness to statistical and hardware-induced noise.

Limitations and Future Work. While HPQS mitigates many challenges of variational quantum learning, its performance depends on the effectiveness of the classical estimator and the quality of quantum measurements. In low-shot regimes, improper blending or weak estimation can limit

improvement over classical baselines. Future directions include learning task-adaptive blending mechanisms, incorporating uncertainty-aware estimators, and extending HPQS to more expressive quantum encodings and structured datasets such as molecular graphs. Furthermore, HPQS can be integrated with quantum error mitigation techniques or classical generative priors, to further enhance performance.

Our work suggests a principled and practical solution for bridging classical and quantum representations in variational quantum learning. It provides a foundation for future hybrid architectures that adaptively combine quantum resources with classical models, without requiring fully quantum inference, which is essential for enabling real-world applications on both noisy intermediate-scale and fault tolerant quantum hardware.

References

[1] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, 2020.

[2] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Physical Review A*, 103(3):032430, 2021.

[3] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Physical review letters*, 122(4):040504, 2019.

[4] Junhua Liu, Kwan Hui Lim, Kristin L Wood, Wei Huang, Chu Guo, and He-Liang Huang. Hybrid quantum-classical convolutional neural networks. *Science China Physics, Mechanics & Astronomy*, 64(9):290311, 2021.

[5] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[6] Diego Ristè, Marcus P Da Silva, Colm A Ryan, Andrew W Cross, Antonio D Córcoles, John A Smolin, Jay M Gambetta, Jerry M Chow, and Blake R Johnson. Demonstration of quantum advantage in machine learning. *npj Quantum Information*, 3(1):16, 2017.

[7] Hsin-Yuan Huang, Michael Broughton, Jordan Cotler, Sitan Chen, Jerry Li, Masoud Mohseni, Hartmut Neven, Ryan Babbush, Richard Kueng, John Preskill, et al. Quantum advantage in learning from experiments. *Science*, 376(6598):1182–1186, 2022.

[8] Marco Cerezo, Guillaume Verdon, Hsin-Yuan Huang, Lukasz Cincio, and Patrick J Coles. Challenges and opportunities in quantum machine learning. *Nature Computational Science*, 2(9):567–576, 2022.

[9] Leonardo Banchi, Jason Pereira, and Marco Zamboni. Few measurement shots challenge generalization in learning to classify entanglement. *arXiv preprint arXiv:2411.06600*, 2024.

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

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

[12] Matthias Christandl and Renato Renner. Reliable quantum state tomography. *Physical Review Letters*, 109(12):120403, 2012.

[13] Zhenyu Cai, Ryan Babbush, Simon C Benjamin, Suguru Endo, William J Huggins, Ying Li, Jarrod R McClean, and Thomas E O'Brien. Quantum error mitigation. *Reviews of Modern Physics*, 95(4):045005, 2023.

[14] Suguru Endo, Simon C Benjamin, and Ying Li. Practical quantum error mitigation for near-future applications. *Physical Review X*, 8(3):031027, 2018.

[15] Daniel A Lidar and Todd A Brun. *Quantum error correction*. Cambridge university press, 2013.

[16] Sergey Bravyi, Andrew W Cross, Jay M Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J Yoder. High-threshold and low-overhead fault-tolerant quantum memory. *Nature*, 627(8005):778–782, 2024.

[17] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, 2017.

[18] Ivan Glasser, Nicola Pancotti, Moritz August, Ivan D Rodriguez, and J Ignacio Cirac. Neural-network quantum states, string-bond states, and chiral topological states. *Physical Review X*, 8(1):011006, 2018.

[19] Michael J Hartmann and Giuseppe Carleo. Neural-network approach to dissipative quantum many-body dynamics. *Physical review letters*, 122(25):250502, 2019.

[20] Kenny Choo, Giuseppe Carleo, Nicolas Regnault, and Titus Neupert. Symmetries and many-body excitations with neural-network quantum states. *Physical review letters*, 121(16):167204, 2018.

[21] Giacomo Torlai, Guglielmo Mazzola, Juan Carrasquilla, Matthias Troyer, Roger Melko, and Giuseppe Carleo. Neural-network quantum state tomography. *Nature physics*, 14(5):447–450, 2018.

[22] Kenny Choo, Antonio Mezzacapo, and Giuseppe Carleo. Fermionic neural-network states for ab-initio electronic structure. *Nature communications*, 11(1):2368, 2020.

[23] Jane Kim, Gabriel Pescia, Bryce Fore, Jannes Nys, Giuseppe Carleo, Stefano Gandolfi, Morten Hjorth-Jensen, and Alessandro Lovato. Neural-network quantum states for ultra-cold fermi gases. *Communications Physics*, 7(1):148, 2024.

[24] James Stokes, Javier Robledo Moreno, Eftychios A Pnevmatikakis, and Giuseppe Carleo. Phases of two-dimensional spinless lattice fermions with first-quantized deep neural-network quantum states. *Physical Review B*, 102(20):205122, 2020.

[25] Samuel Yen-Chi Chen, Chao-Han Huck Yang, Jun Qi, Pin-Yu Chen, Xiaoli Ma, and Hsi-Sheng Goan. Variational quantum circuits for deep reinforcement learning. *IEEE access*, 8:141007–141024, 2020.

[26] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. Expressive power of parametrized quantum circuits. *Physical Review Research*, 2(3):033125, 2020.

[27] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized quantum circuits as machine learning models. *Quantum science and technology*, 4(4):043001, 2019.

[28] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *nature*, 549(7671):242–246, 2017.

[29] Lorenzo Leone, Salvatore FE Oliviero, Lukasz Cincio, and Marco Cerezo. On the practical usefulness of the hardware efficient ansatz. *Quantum*, 8:1395, 2024.

[30] Marcus Cramer, Martin B Plenio, Steven T Flammia, Rolando Somma, David Gross, Stephen D Bartlett, Olivier Landon-Cardinal, David Poulin, and Yi-Kai Liu. Efficient quantum state tomography. *Nature communications*, 1(1):149, 2010.

[31] Ming-Chien Hsu, En-Jui Kuo, Wei-Hsuan Yu, Jian-Feng Cai, and Min-Hsiu Hsieh. Quantum state tomography via nonconvex riemannian gradient descent. *Physical Review Letters*, 132(24):240804, 2024.

[32] Abhinav Kandala, Kristan Temme, et al. Error mitigation extends the computational reach of a noisy quantum processor. *Nature*, 567:491–495, 2019.

[33] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[34] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[35] Chen-Yu Liu, En-Jui Kuo, Chu-Hsuan Abraham Lin, Jason Gemsun Young, Yeong-Jar Chang, Min-Hsiu Hsieh, and Hsi-Sheng Goan. Quantum-train: Rethinking hybrid quantum-classical machine learning in the model compression perspective. *arXiv preprint arXiv:2405.11304*, 2024.

[36] Chen-Yu Liu, En-Jui Kuo, Chu-Hsuan Abraham Lin, Sean Chen, Jason Gemsun Young, Yeong-Jar Chang, and Min-Hsiu Hsieh. Training classical neural networks by quantum machine learning. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 2, pages 34–38. IEEE, 2024.

[37] Chen-Yu Liu, Chu-Hsuan Abraham Lin, Chao-Han Huck Yang, Kuan-Cheng Chen, and Min-Hsiu Hsieh. Qtrl: Toward practical quantum reinforcement learning via quantum-train. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 2, pages 317–322. IEEE, 2024.

[38] Chu-Hsuan Abraham Lin, Chen-Yu Liu, and Kuan-Cheng Chen. Quantum-train long short-term memory: Application on flood prediction problem. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 2, pages 268–273. IEEE, 2024.

[39] Chen-Yu Liu, Chu-Hsuan Abraham Lin, and Kuan-Cheng Chen. Quantum-train with tensor network mapping model and distributed circuit ansatz. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–4. IEEE, 2025.

[40] Hanrui Wang, Yongshan Ding, Jiaqi Gu, Yujun Lin, David Z Pan, Frederic T Chong, and Song Han. Quantumnas: Noise-adaptive search for robust quantum circuits. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 692–708. IEEE, 2022.

[41] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[42] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[43] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[44] Chen-Yu Liu, Chao-Han Huck Yang, Hsi-Sheng Goan, and Min-Hsiu Hsieh. A quantum circuit-based compression perspective for parameter-efficient learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[45] Chen-Yu Liu and Samuel Yen-Chi Chen. Federated quantum-train with batched parameter generation. In *2024 15th International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1133–1138. IEEE, 2024.

[46] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[47] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[48] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[49] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. *Advances in neural information processing systems*, 29, 2016.

[50] Jacob Miller. Torchmps. https://github.com/jemisjoky/torchmps, 2019.

[51] 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.

[52] Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, and Tristan Cook. Quanvolutional neural networks: powering image recognition with quantum circuits. *Quantum Machine Intelligence*, 2(1):2, 2020.

[53] Andrea Mari, Thomas R Bromley, Josh Izaac, Maria Schuld, and Nathan Killoran. Transfer learning in hybrid classical-quantum neural networks. *Quantum*, 4:340, 2020.

[54] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised learning with quantum enhanced feature spaces. *Nature*, 567(7747):209–212, 2019.

[55] Armands Strikis, Dayue Qin, Yanzhu Chen, Simon C Benjamin, and Ying Li. Learning-based quantum error mitigation. *PRX Quantum*, 2(4):040330, 2021.

[56] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout Van Den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, et al. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

A Effects of Quantum Computer Noise

Table 4: Comparison of QML models under different noise settings. (shot count $= 20\times$ HSS)

QML (MNIST-2)			
Model	**Testing Accuracy (%)**	**# Training Param.**	**Noise Model**
HPQS (finite)	88.97 ± 5.08	65	—
HPQS (finite)	$\mathbf{93.33 \pm 1.08}$	65	ibm_fez
HPQS (finite)	$\mathbf{90.21 \pm 3.50}^{\dagger}$	65	ibm_torino
PQC (finite)	53.33 ± 6.61	40	—
PQC (finite)	44.44 ± 4.12	40	ibm_fez
PQC (finite)	55.10 ± 6.65	40	ibm_torino

Table 5: Comparison of QT models under different noise settings. (shot count $= 10\times$ HSS)

QT (MNIST-10)			
Model	**Testing Accuracy (%)**	**# Training Param.**	**Noise Model**
HPQS (finite)	86.28 ± 0.58	649	—
HPQS (finite)	77.04 ± 7.79	649	ibm_fez
HPQS (finite)	79.58 ± 4.88	649	ibm_torino
HPQS (finite)	$\mathbf{87.64 \pm 0.47}$	792	—
HPQS (finite)	$\mathbf{87.04 \pm 0.52}^{\dagger}$	792	ibm_fez
HPQS (finite)	81.62 ± 4.58	792	ibm_torino
PQC (finite)	57.54 ± 5.61	1164	—
PQC (finite)	61.94 ± 5.49	1164	ibm_fez
PQC (finite)	56.11 ± 2.87	1164	ibm_torino
PQC (finite)	65.83 ± 2.56	1424	—
PQC (finite)	55.89 ± 4.39	1424	ibm_fez
PQC (finite)	56.82 ± 4.07	1424	ibm_torino

Table 6: Comparison of QPA models (GPT-2) under different noise settings. (shot count $= 1\times$ HSS)

QPA (GPT-2 & Wikitext-2)			
Model	**Testing PPL**	**# Training Param.**	**Noise Model**
HPQS (finite)	$\mathbf{4.612}$	23585	—
HPQS (finite)	53.565	23585	ibm_fez
HPQS (finite)	53.785	23585	ibm_torino
PQC (finite)	$\mathbf{6.975}^{\dagger}$	22368	—
PQC (finite)	86.695	22368	ibm_fez
PQC (finite)	88.369	22368	ibm_torino

To assess the noise robustness of HPQS, we evaluate its performance under realistic quantum noise simulation using IBM's available noise models [2] ibm_torino and ibm_fez. Across three representative settings: expectation-based QML, QT, and QPA. Each task is executed with a fixed shot budget, specifically $20\times$ HSS for QML, $10\times$ HSS for QT, and $1\times$ HSS for QPA.

Expectation-based QML HPQS under Noise setting. As shown in Table 4, HPQS maintains high classification accuracy under both noise-free and noisy conditions. For instance, it achieves 93.33% accuracy under ibm_fez and 90.21% under ibm_torino, both of which surpass the corresponding PQC (finite) results that degrade to 44.44% and 55.10%, respectively. Notably, HPQS also outperforms the PQC (exact) baseline (86.66%) in the noise-free case.

QT with HPQS under Noise setting. In the QT scenario (Table 5), HPQS achieves 87.64% testing accuracy without noise and retains strong performance under noisy settings (87.04% with ibm_fez and 81.62% with ibm_torino). In contrast, PQC (finite) experiences significant degradation, dropping to accuracies between 55% and 65% depending on noise level and parameter count.

---

[2] IBM Quantum provides noise models based on the properties of real hardware backends: https://docs.quantum.ibm.com/api/qiskit/0.19/qiskit.providers.aer.noise.NoiseModel.

Table 7: Comparison of QPA models (Gemma-2) under different noise settings. (shot count = $1\times$ HSS)

QPA (Gemma-2 & Wikitext-2)			
Model	**Testing PPL**	**# Training Param.**	**Noise Model**
HPQS (finite)	**1.467**	141585	—
HPQS (finite)	**1.470**[†]	141585	ibm_fez
HPQS (finite)	1.472	141585	ibm_torino
PQC (finite)	1.472	140432	—
PQC (finite)	1.472	140432	ibm_fez
PQC (finite)	1.472	140432	ibm_torino

QPA with HPQS under Noise setting. For QPA (Table 6 and Table 7), which targets LoRA-based fine-tuning of large language models, HPQS consistently achieves lower perplexity than PQC (finite) under both noise-free and noisy conditions. In the GPT-2 experiment, HPQS yields 4.612 in the noise-free setting, while the noisy variants (ibm_fez and ibm_torino) reach 53.565 and 53.785, respectively. These values remain substantially better than PQC (finite), which suffers from notable degradation, reaching 86.695 to 88.369 under noise. A similar trend is observed in the Gemma-2 experiment, where HPQS achieves 1.467 to 1.472 perplexity across all conditions, marginally outperforming PQC (finite).

B   Implementation Details and Training Hyperparameter Configuration

In this section, we provide the training hyperparameter configuration used for the results presented in the main text. All experiments were conducted on a single NVIDIA V100S GPU with 32GB VRAM.

B.1   Tensor Network Mapping via Matrix Product States

Tensor networks (TNs) have emerged as powerful tools for modeling high-dimensional correlations in quantum many-body physics, classical machine learning, and quantum circuit simulations. Their compact and structured representation makes them especially well-suited for capturing the relationship between quantum measurement distributions and classical neural network parameters.

Following the supervised learning formulation of matrix product states (MPS) in [49], we parameterize the mapping function from quantum-classical inputs to model weights using a tensor decomposition. Specifically, the output vector $\vec{\omega} \in \mathbb{R}^d$ is computed via a contraction between an MPS weight tensor W and an input-dependent feature map $\Xi(\mathbf{x})$, where:

$$W_{s_1, s_2, \ldots, s_{N+1}} = \sum_{\boldsymbol{\alpha}} A^{\alpha_1} s_1 A^{\alpha_1 \alpha_2} s_2 \cdots A^{\alpha_N} s_{N+1}, \tag{20}$$

and $\mathbf{x} = (x_1, x_2, \ldots, x_{N+1}) \in \mathbb{R}^{N+1}$ represents the concatenated input composed of the computational basis state $|\phi\rangle$ and the associated measurement probability $|\langle \phi | \psi(\boldsymbol{\theta}) \rangle|^2$. The corresponding feature map is expressed as:

$$\Xi^{s_1, s_2, \ldots, s_{N+1}}(\mathbf{x}) = \xi^{s_1}(x_1) \otimes \xi^{s_2}(x_2) \otimes \cdots \otimes \xi^{s_{N+1}}(x_{N+1}), \tag{21}$$

where each local map $\xi^{s_j}(x_j) \in \mathbb{R}^2$ is defined by:

$$\xi^{s_j}(x_j) = \begin{bmatrix} x_j \\ 1 - x_j \end{bmatrix}. \tag{22}$$

The final output of the tensor network is obtained as:

$$G(\mathbf{x}) = W \cdot \Xi(\mathbf{x}) = \vec{\omega}. \tag{23}$$

Here, the MPS core tensors $A_{s_j}^{\alpha_j}$ are the learnable parameters of the model, with bond dimension r governing the expressivity. This approach allows for structured generalization and efficient parameter scaling in the mapping of hybrid quantum-classical features to downstream model components.

Returning to Eq. 15 in the main text, the tensor network mapping functions applied to the quantum and classical branches are expressed as:

$$\tilde{a}_i^{(Q)} = G_{\text{TN}}(\phi_i, \hat{p}_i^{\text{emp}}), \quad \tilde{a}_i^{(C)} = H_{\text{TN}}(\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)). \tag{24}$$

These tensor network functions are structurally aligned with the contraction form presented in Eq. 23, where the resulting output vector $\vec{\omega}$ corresponds to either $\tilde{a}_i^{(Q)}$ or $\tilde{a}_i^{(C)}$, depending on whether it originates from the quantum or classical branch. In our implementation, the bond dimensions are task-specific: for the Quantum-Train experiments, the quantum-side tensor network G_{TN} is assigned a bond dimension of $r = 2$, while the classical-side H_{TN} adopts a minimal structure with $r = 1$. In contrast, for the QPA experiments involving large-scale language models, we employ a more expressive configuration with $r = 10$ for G_{TN} and $r = 4$ for H_{TN}. These mapping modules are implemented using the `TorchMPS` library [50], which provides efficient support for matrix product state (MPS) computations in PyTorch.

B.2 Parameter Settings for Expectation-Based QML Experiments

We evaluate HPQS in an expectation-based QML task, closely following the setup in [40]. The quantum branch utilizes a 4-qubit PQC, corresponding to a HSS of $2^4 = 16$. Input MNIST images are downsampled to 4×4 using PyTorch's avg_pool2d operation. The resulting 16-dimensional vectors are mapped to gate angles using a sequence of rotation gates: 4RY, 4RZ, 4RX, and 4RY. Measurements are performed in the Pauli-Z basis, producing expectation values in $[-1, 1]$ for each qubit. For binary classification (digits 3 vs 6), the expectation values of qubits $\{0, 1\}$ and $\{2, 3\}$ are summed independently to form two logits, which are then normalized using a softmax function to yield class probabilities (corresponding to postprocessing G).

In the classical branch (NQS), the same 4×4 image is reshaped into a 16-dimensional vector and passed through a fully connected layer of shape $(16, 1)$, followed by a ReLU activation and a final linear layer of shape $(1, 2)$ to produce logits. The quantum and classical outputs are blended using a fixed coefficient $\lambda = 0.1$. The entire hybrid model is trained using the negative log-likelihood (NLL) loss over 5 epochs, optimized with Adam at a learning rate of 5×10^{-3}. Each reported result is averaged over three random seeds.

B.3 Parameter Settings for Quantum-Train Experiments

For the QT task, we adopt a setup aligned with the original framework proposed by [35]. The objective is to generate the full parameter set of a compact CNN comprising 6690 trainable parameters. To match this capacity, the quantum branch uses $n = \lceil \log_2 6690 \rceil = 13$ qubits, resulting in a HSS of $2^{13} = 8192$. Unlike the expectation-based QML setting, the classification task here spans all 10 MNIST digit classes. Full 28×28 grayscale images are used as input to the CNN. The quantum state is prepared using a layered parameterized ansatz defined as:

$$|\psi(\boldsymbol{\theta})\rangle = \left(\prod_{i=1}^{n-1} \text{CNOT}^{i,i+1} \prod_{j=1}^{n} R_Y^j(\theta_j^{(L)}) \right)^L |0\rangle^{\otimes n}, \tag{25}$$

where $L = 1$ denotes the number of layers, each composed of entangling CNOT gates and single-qubit rotations.

The classical branch (NQS) processes the binary representation of each basis state $\phi_i \in \{0, 1\}^n$ through a fully connected layer of shape $(n, 32)$, followed by a ReLU activation and a final linear layer of shape $(32, 1)$, yielding a scalar probability estimate. The quantum and classical branches are blended using a fixed coefficient $\lambda = 0.5$. Training is performed for 50 epochs using the Adam optimizer with a learning rate of 1×10^{-4}. The loss function is categorical cross-entropy, as appropriate for multi-class classification. All results are averaged over three random seeds for statistical reliability.

B.4 Parameter Settings for Quantum Parameter Adaptation

For the QPA experiments, we largely follow the setup proposed in the original QPA work [44], as summarized in Table 8. In this setting, α denotes the scaling factor used in the low-rank adaptation (LoRA) scheme. The chunk size for batched parameter generation is set to $n_{\text{mlp}} = 512$ for GPT-2 and $n_{\text{mlp}} = 4096$ for Gemma-2. Given this chunking strategy, the required number of qubits is determined

by (Eq. 19):

$$n_{\text{GPT-2}} = \left\lceil \log_2 \left\lceil \frac{204100}{512} \right\rceil \right\rceil = 9, \tag{26}$$

$$n_{\text{Gemma-2}} = \left\lceil \log_2 \left\lceil \frac{1032192}{4096} \right\rceil \right\rceil = 8, \tag{27}$$

where 204100 and 1032192 are the number of trainable parameters in the LoRA module ($r = 4$) for the final linear layers of GPT-2 and Gemma-2, respectively. The quantum ansatz employed in this setting mirrors the layered structure used in the QT experiments, with a circuit depth of $L = 8$. The classical NQS branch adopts the same architecture as in the QT setup, consisting of a fully connected network that operates on the binary representation of basis states. The outputs of the quantum and classical branches are combined using a fixed blending coefficient of $\lambda = 0.5$.

Table 8: Hyperparameter configurations of QPA LoRA for fine-tuning GPT-2 and Gemma-2 with WikiText-2 dataset.

| Hyperparameters | QPA LoRA | |
	GPT-2	Gemma-2
LoRA Rank r	4	
α	2r	
Dropout	0.05	0.0
Optimizer	AdamW	
LR	1e-5	
LR Scheduler	Linear	
Batch size	1	
Warmup Steps	0	
Epochs	3	

C Related Quantum Learning Frameworks as Special Cases of HPQS

The HPQS framework defines a hybrid variational quantum model by blending postprocessed outputs from a quantum circuit and a classical neural estimator:

$$\hat{p}_i = \lambda \cdot G(\phi_i, \hat{p}_i^{\text{emp}}) + (1 - \lambda) \cdot H(\phi_i, f_{\boldsymbol{\gamma}}(\phi_i)), \tag{28}$$

where:

- $\hat{p}_i^{\text{emp}}$ is the empirical probability estimate from quantum measurements
- $f_{\boldsymbol{\gamma}}(\phi_i)$ is the NQS-predicted value
- $G(\cdot)$ and $H(\cdot)$ are postprocessing functions (can be both quantum or classical)
- $\lambda \in [0, 1]$ controls the contribution of each branch

This formulation naturally subsumes a range of quantum learning models:

PQC-Only Learning and NQS-Only Learning.   As discussed extensively in the main text, HPQS recovers conventional variational quantum models in the limiting case of $\lambda = 1$ with a simple postprocessing function G. This corresponds to purely PQC-based learning, such as in variational quantum algorithms (VQAs) [51], where optimization relies entirely on quantum circuit evaluations. Conversely, when $\lambda = 0$ and simple H, HPQS reduces to a NQS formulation [17]. These limiting cases highlight the generality of HPQS as a unifying framework for both purely quantum and purely classical learning paradigms.

Hybrid Quantum-Classical Learning.   Several models in QML apply classical neural layers directly on top of quantum features. For instance, Quanvolutional Neural Networks [52] use quantum circuits as feature extractors followed by classical classifiers. These can be seen as cases where $G(\cdot)$ includes a learnable classical transformation and $\lambda = 1$.

The framework introduced in [53] appends PQCs after classical neural network layers, effectively enabling transfer learning from classical to quantum domains. Within the HPQS formulation, this can be interpreted as a special case where the blending coefficient is set to $\lambda = 0$, and the postprocessing function in the classical branch $H(\cdot)$ includes a learnable quantum circuit applied to the classical output. This highlights how quantum layers can act as task-adaptive decoders within the broader HPQS architecture.

Quantum-Classical Embeddings. Recent work on quantum-enhanced feature spaces, such as Quantum Support Vector Machines [54], implicitly postprocess quantum features via classical similarity or metric functions. These frameworks correspond to HPQS with fixed or partially learnable G and $\lambda = 1$.

Classical Postprocessing and Noise Mitigation. Classical estimators are increasingly used to compensate for quantum noise. Notable examples include error mitigation via neural networks and other approaches [55, 56]. These methods naturally fit into the HPQS architecture as specific instantiations of $G(\cdot)$, and highlight the flexibility of HPQS in NISQ settings.

HPQS provides a general and expressive framework that unifies diverse quantum-classical learning paradigms. Its modular structure supports flexible combinations of classical and quantum components, with the blending coefficient λ offering explicit control over their respective contributions. The incorporation of postprocessing functions G and H further enhances the adaptability of each branch to task-specific requirements. While the current formulation considers a fixed λ across the entire system, future extensions may explore *context-aware* λ_i values, allowing the model to dynamically adjust the quantum-classical trade-off per input or per output component. This offers a promising direction for developing more adaptive and data-efficient hybrid architectures.

D Why Increasing Qubit Count Leads to Higher Estimation Error

We extend the analysis in the main text to formally characterize how measurement uncertainty scales with the number of qubits. Let $\hat{P}(|\phi_i\rangle)$ denote the empirical estimate of the probability of observing basis state $|\phi_i\rangle$, based on n'_{shot} measurement repetitions. Here, $n'_{\text{shot}} = n_{\text{shot}}/2^n$ represents the average number of shots allocated per basis state in an n-qubit system with Hilbert space size 2^n.

Hoeffding's inequality provides a probabilistic bound on the deviation of this empirical estimate from its true expectation:

$$P\left(\left|\hat{P}(|\phi_i\rangle) - \mathbb{E}[P(|\phi_i\rangle)]\right| \geq \epsilon\right) \leq 2\exp(-2\epsilon^2 n'_{\text{shot}}). \tag{29}$$

Rewriting the bound in terms of a fixed confidence level $\delta \in (0,1)$, we obtain:

$$\epsilon = \sqrt{\frac{1}{2n'_{\text{shot}}}\ln\left(\frac{2}{\delta}\right)} = \sqrt{\frac{2^n}{2n_{\text{shot}}}\ln\left(\frac{2}{\delta}\right)}. \tag{30}$$

This reveals a key limitation: under a fixed total shot budget n_{shot}, the estimation error ϵ grows exponentially with the number of qubits n. In other words, as the Hilbert space expands, fewer measurements are allocated per basis state, leading to larger statistical fluctuations. This scaling behavior underscores the importance of shot-efficient designs, such as HPQS, which integrate classical estimators to reduce reliance on dense quantum measurements.