IEEE Transactions on Quantum Engineering

Received 8 February 2025; revised 7 November 2025; accepted 25 November 2025; date of publication 1 December 2025; date of current version 31 December 2025.

Digital Object Identifier 10.1109/TQE.2025.3638878

Relative Entropy-Based Training of Quantum Neural Networks

SUBHADEEP MONDAL (Graduate Student Member, IEEE), AND AMIT KUMAR DUTTA

G.S. Sanyal School of Telecommunication, Indian Institute of Technology Kharagpur, Kharagpur 721302, India

Corresponding author: Amit Kumar Dutta (e-mail: amitkdutta@yahoo.co.in).

This work was supported by Science and Engineering Research Board, Govt. of India under Project CRG2023004209.

ABSTRACT Quantum neural networks (QNNs) are gaining attention as versatile models for quantum machine learning, but training them effectively remains a challenge. Most existing approaches, such as quantum multilayer perceptrons, use fidelity-based cost functions. While well-suited for pure states, these measures are less reliable when inputs and outputs are mixed states—a situation common in learning quantum channels. In this work, we introduce a training framework built on a relative entropy-inspired cost function. By quantifying the directional divergence between learned and target states, relative entropy provides a more informative and principled measure than linear fidelity, naturally capturing both spectral and eigenvector differences in mixed states. This approach preserves the completely positive structure of the network, supports efficient backpropagation in layered QNN configurations, and achieves improved accuracy and convergence over fidelity-based training. These results highlight entropy-based optimization as a promising path toward scalable, robust, and noise-resilient quantum learning.

INDEX TERMS Decoherence noise, quantum channel, quantum mixed state learning, quantum neural networks (QNNs), quantum relative entropy.

NOMENCLATURE

Notations: The notations used in this work are given in Nomenclature.

Symbol	Description
σ_x^{in}	Input state to the QNN for sample index x.
σ_x^{out}	Predicted QNN output state for sample index x.
σ_x^l	Intermediate state at layer l for sample index x.
ρ_x	Target state corresponding to the desired output.
W_p^l	Unitary to the pth perceptron in the lth layer.
V_p^l	Parameter matrix used to update W_p^l.
k_l	Number of perceptrons coupling layer in $(l-1)$ to layer l.
Tr(.)	Trace of a matrix.
$\mathcal{T}^l$	Forward positive transition map at layer l.
$\mathcal{G}^l$	Adjoint completely positive (CP) map at layer l.
B^l	Backward-propagated operator at the lth layer.
H_x	Density-like operator appearing in the adjoint gradient channel for sample x.
D_x^l	Intermediate forward-propagated state appearing in the construction of B^l.
Q_x^l	Contribution of layer l to the overall gradient signal for sample index x.
S_x	Relative entropy (loss) between the predicted and target states for sample x.
x	Index of the training data, $x \in \{1, \ldots, N\}$.
δ	Learning step size used for unitary updates.
t	Iteration step in the training procedure.

I. INTRODUCTION

Quantum machine learning (QML) combines quantum computing with machine learning to address complex problems more efficiently than classical methods, utilizing phenomena like superposition and entanglement. Various QML models have been developed, such as Born quantum circuit machine, quantum generative adversarial network, and Boltzmann quantum machine [1], [2], [3], each leveraging quantum computing in unique ways for tasks like generative modeling and adversarial training.

Quantum neural networks (QNNs) leverage quantum principles to enhance computational capabilities, inspired by classical CNNs [4]. Early adaptations include parameterized quantum circuits and quantum single-layer perceptrons [5], [6], which paved the way for configurations like QDCNNs and QCNNs [7], [8]. Pure quantum models face scalability challenges due to quantum random-access memory (QRAM) requirements and costly procedures like quantum phase estimation [9]. Hybrid quantum-classical approaches address

these limitations. Henderson et al. [10] introduced Quanvolutional layers, embedding quantum circuits into classical CNNs, while Kerenidis et al. [11] implemented quantum convolution using linear algebra routines with QRAM. Over-parameterization and effective dimension studies indicate strong training capacity and generalization [12], [13], and QNNs remain robust under noisy intermediate-scale quantum (NISQ) noise [14]. Even without hardware noise, challenges like barren plateaus and local minima persist [15], [16], motivating quantum analogues of classical components—perceptrons, configurations, optimizers, and loss functions [17], [18]. Classical-inspired initialization in VQAs helps mitigate barren plateaus [19], yet defining the optimal "quantum perceptron" remains open [20]. Quantum perceptrons, developed in both gate-model and adiabatic quantum computing frameworks, have given us important insights into the capabilities and implementations of QNNs [21].

Motivated by these advances, researchers have explored using QNNs to learn entire quantum channels, capturing not only unitary dynamics but also the effects of noise and decoherence. Building on this idea, Beer et al. [22] proposed fully quantum deep neural networks capable of universal quantum computation, laying the groundwork for such channel-learning applications. These quantum perceptrons are trained using the adjoint-channel (adjoint differentiation) method, which computes gradients through a forward pass to store intermediate states followed by a backward pass that applies the adjoint of each layer, in contrast to the parameter-shift rule [23], which requires repeated circuit evaluations for each parameter. The algorithm is a quantum analog of classical multilayer perceptrons (MLPs), structuring quantum perceptrons as unitary operators organized into layers and interconnected via layerwise qubits, which are discarded layer by layer after interactions. Moreover, the perceptron unitaries are initialized randomly, and the network architecture is constructed such that the overall training process remains efficient, depending only on the width of individual layers rather than on the total network depth. This structural property helps to avoid the exponential scaling associated with the underlying Hilbert space, thereby making gradient-based optimization practically realizable. When trained with a fidelity-based loss function, the proposed QNN preserves the universal approximation capability analogous to that of classical MLPs. A similar QNN architecture has also been implemented on a superconducting processor [24], demonstrating its experimental feasibility. Furthermore, the quantum information bottleneck (QIB) framework introduced by Çatlıand Wiebe [25] provided an information-theoretic perspective on QNN training. In QIB, the learning objective is formulated to maximize the relevant information captured by the network while compressing irrelevant components of the input quantum states. This approach formalizes the trade-off between accuracy and information compression in QNNs and has inspired subsequent frameworks for structured, resource-efficient training of quantum networks.

However, when the input and output states are mixed states, the QNN must be trained directly on such mixed-state training pairs. This situation is not merely a theoretical possibility but arises naturally in practical QML tasks, particularly when the objective is to learn an entire quantum channel rather than just a unitary transformation, as also discussed in [22]. In realistic scenarios, quantum channels are rarely noise-free; they typically model open-system dynamics where interaction with the environment introduces decoherence and dissipation [26]. As a consequence, both the input and output states of the channel are described by density matrices rather than pure states [27]. Learning such channels is therefore a fundamentally mixed-state problem. Training with mixed states, however, introduces unique challenges. A central difficulty lies in the choice of the cost function. Fidelity-based measures are highly effective in pure-state settings because the overlap between states provides a clear operational meaning and a straightforward optimization landscape. Yet, in mixed-state contexts, fidelity behaves differently [28]: two mixed states can exhibit high fidelity even when their underlying distributions of eigenvalues differ substantially, and conversely, small perturbations due to noise may significantly alter the fidelity value. This makes it less straightforward to apply as a training metric for QNNs intended to model noisy or open-system dynamics. To overcome these limitations, our proposed work introduces a training algorithm that optimizes a relative-entropy-based cost function [29]. Entropic quantities are particularly well-suited for mixed-state scenarios because they capture not only the overlap of states but also their informational distinguishability. Relative entropy, in particular, quantifies how well one density matrix approximates another and has a direct operational interpretation in terms of hypothesis testing. Moreover, entropy-based measures remain robust in the presence of noise and randomness, which are inherent to mixed-state quantum data [30]. This makes them a natural and principled choice for channel learning tasks. It is worth emphasizing that our algorithm does not rely on any special assumptions about the structure of the mixed states themselves. Instead, it adapts the QNN architecture so that the entropy-based cost can be efficiently evaluated during both feed-forward and backpropagation stages. In this way, the framework remains fully general while extending the reach of QNN training to mixed state training pairs. Such developments mark an important step toward realizing the full potential of QML, as they provide the tools to handle the complexities of quantum channels.

Contributions: Given the above context, the main contributions of this work are outlined below.

1) *Relative-entropy cost formulation for mixed-state QNNs:* We propose a training objective based on quantum relative entropy, designed for mixed-state training pairs in QNNs [22]. Unlike fidelity-based metrics, which primarily measure state overlap and may lose sensitivity under noise and decoherence, this cost

function captures the divergence between the learned and target states, providing a more informative way to guide training for realistic mixed-state QNNs.

2) *Entropy-aware backpropagation under CP-map constraints:* We develop a modified adjoint-channel backpropagation framework that incorporates the relative-entropy cost function while enforcing complete positivity of the QNN layers. The gradient update rule is equipped with a controlled step-size schedule adapted to the sharper sensitivity of entropy-based training, resulting in stable and smooth convergence dynamics.
3) *Systematic architecture evaluation with CI-based generalization:* We conduct a systematic comparison across a family of shallow and moderately deep QNN configurations and find that width-enhanced shallow models achieve superior convergence efficiency, consistent with recent observations in quantum neural design principles. Generalization capability is quantified using statistically rigorous 95% confidence intervals (CIs) derived from k-fold cross-validation across multiple independent channel realizations, ensuring reproducibility and robustness of the reported trends.
4) *Noise-resilient optimization validated under depolarizing channels:* Beyond idealized training conditions, we empirically demonstrate that the entropy-based optimization retains convergence stability even when input states are perturbed by CPTP depolarizing noise maps. The observed controlled degradation under increasing noise strength confirms that the proposed QNN framework is inherently compatible with NISQ-era decoherence profiles, without requiring any modification to the learning rule.

The rest of this article is organized as follows. Section I introduces the motivation behind mixed-state QNN training and highlights the limitations of fidelity-based objectives. Section II presents the proposed relative-entropy–based QNN training framework, detailing the architecture, cost formulation, adjoint backpropagation under completely positive trace preserving (CPTP) constraints, comparison with the QIB approach, and postprocessing with complexity analysis. Section III reports the simulation setup and results across different QNN configurations, including cross-validation performance, CI-based generalization analysis, and robustness under depolarizing noise. Finally, Section IV summarizes the key contributions and outlines future research directions.

II. PROPOSED METHODOLOGY

We now describe the background of the existing QNN method and its corresponding cost function followed by our proposed one.

A. EXISTING QNN ARCHITECTURE

A QNN operates on a set of training inputs $\{\sigma_x^{\text{in}}\}$ and produces the corresponding output states $\{\sigma_x^{\text{out}}\}$, which in general may be mixed due to entanglement with ancillary subsystems and the partial trace over intermediate layers. The evolution of the QNN is described as a composition of CP transition maps $\mathcal{T}^l$, each representing the action of a single layer on the quantum state. Accordingly, the overall network transformation for a given input σ_x^{in} is written as

$$\sigma_x^{\text{out}} = \mathcal{T}^{\text{out}}\left(\mathcal{T}^k\left(\ldots\left(\mathcal{T}^2\left(\mathcal{T}^1\left(\sigma_x^{\text{in}}\right)\right)\right)\ldots\right)\right) \tag{1}$$

where the CP maps $\mathcal{T}^l$ are applied sequentially from the input layer to the output layer, consistent with the feedforward QNN framework introduced in Beer et al. [22]. At the level of an individual layer, the transformation takes the form

$$\mathcal{T}^l\left(\sigma_x^{l-1}\right) = \text{Tr}_{l-1}\left[\left(\prod_{p=1}^{k_l} W_p^l\right)\left(\sigma_x^{l-1}\otimes|0\cdots0\rangle\langle0\cdots0|\right)\times\left(\prod_{p=1}^{k_l} W_p^l\right)^{\dagger}\right] \tag{2}$$

where W_p^l denotes the unitary associated with the pth perceptron in layer l, and k_l is the total number of perceptrons coupling layer $(l-1)$ to layer l. Each W_p^l acts nontrivially on the qubits of its preceding layer together with the qubit representing the perceptron in the current layer, while operating as the identity on all remaining subsystems. Each layer therefore processes its input state together with ancillas, applies the sequence of perceptron unitaries, and discards the previous layer's qubits via partial trace. Each layer thus processes $\sigma_{l-1,x}$ together with ancilla qubits initialized in $|0\cdots0\rangle$, applies a sequence of perceptron unitaries, and discards the qubits of the previous layer via partial trace. This formalism provides a transparent representation of the layered architecture and highlights how mixed states arise naturally within the QNN structure.Quantum perceptron forms the building block for the QNN architecture, which consists of L hidden layers of qubits as shown in Fig. 1.

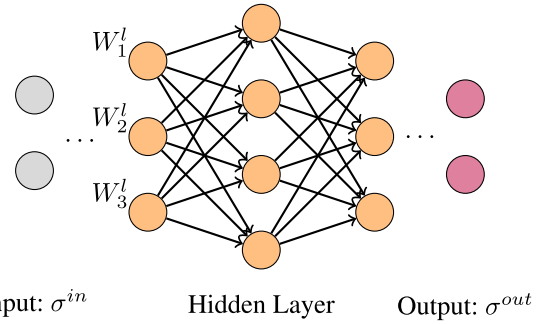


FIGURE 1. Diagram of the QNN architecture.

The analogy with classical feed-forward neural networks is clear: information propagates forward through successive layers of perceptrons. Crucially, this architecture also admits a natural mechanism for backpropagation. As discussed in

Beer et al. [22], the forward propagation is governed by the CP maps $\mathcal{T}^l$ on the input, while the backward propagation during training is carried out by their corresponding adjoint channels on a postprocessed state (depends on the predicted and target output). This duality provides a genuine quantum analogue of classical backpropagation, with the advantage that optimization can be performed layer by layer without requiring access to the full global unitary of the network.

To train a QNN, a cost function is required to quantify how closely the output matches the desired target. For pure input and target states, Beer et al. [22] used a fidelity-based cost function

$$C_{fid} = \frac{1}{N}\sum_x \langle \psi_{\text{out}} | \sigma^{\text{out}} | \psi_{\text{out}} \rangle \tag{3}$$

which parallels classical risk functions and is suitable for learning tasks where both input and target remain pure, such as unknown unitary operations. When the input σ^{in} or target ρ is mixed, for instance due to noise or open-system dynamics, the appropriate figure of merit is the Uhlmann fidelity

$$F\left(\rho, \sigma^{\text{out}}\right) = \left(\text{Tr}\left(\sqrt{\sqrt{\rho}\,\sigma^{\text{out}}\,\sqrt{\rho}}\right)\right)^2. \tag{4}$$

While (4) reduces to the pure-state overlap in the limit of pure states, it remains valid in the presence of mixedness and decoherence. Notably, existing QNN training approach [22] based on analytic gradients were derived using (3) and do not directly optimize this more general mixed-state cost, highlighting the need for a training objective that properly accounts for mixed-state dynamics.

B. PROPOSED QNN COST FUNCTION

Building on the discussion of mixed-state fidelity, a natural next step is to develop a training objective that is inherently suited for mixed-state scenarios. To this end, we propose a cost function based on quantum relative entropy, which provides a principled measure of divergence between the target state and the QNN output. Unlike fidelity in (3), which may lose sensitivity in noisy or decohered states, the relative-entropy objective is well-suited for learning with mixed-state pairs and extends the applicability of QNN training to more general, realistic quantum channels.

Assume that there are N training states, indexed by x, where $x \in [1, \ldots, N]$. The average relative entropy-based cost function evaluates the difference between the predicted output state (σ_x^{out}) and the corresponding target state (ρ_x) for the xth training instance. This cost function provides an overall measure of the QNN's performance across all training states and is defined as

$$C = \frac{1}{N}\sum_{x=1}^{N} S\left(\rho_x \| \sigma_x^{\text{out}}\right) \tag{5}$$

where $S(\rho_x \| \sigma_x^{\text{out}})$ represents the relative entropy [29] between the target state ρ_x and the predicted output state σ_x^{out} for the xth training state, with the quantum relative entropy defined as

$$S\left(\rho_x \| \sigma_x^{\text{out}}\right) = \text{Tr}\left(\rho_x \ln \rho_x - \rho_x \ln \sigma_x^{\text{out}}\right). \tag{6}$$

The primary objective of this framework is to *minimize* the quantum relative entropy $S(\rho_x \| \sigma_x^{\text{out}})$ between the predicted state σ_x^{out} and the target state ρ_x, as it quantifies their statistical distinguishability and drives the QNN training process.

In this framework, we describe the minimization of the relative entropy cost function through layer-wise optimization of the perceptron parameters $V_p^l(t)$, which update the perceptron unitaries $W_p^l(t)$ in the QNN. The overall training procedure is summarized in Algorithm 1. For each infinitesimal time increment $t \leftarrow t + \delta$, the objective is to minimize the cost function $C(t)$, explicitly dependent on the time evolution parameter t. The derivative of the relative entropy cost function is expressed as

$$\begin{aligned}\frac{dC(t)}{dt} &= -\frac{1}{N}\sum_{x=1}^{N} \text{Tr}\left[\rho_x (\sigma_x^{\text{out}}(t))^{-1} \frac{d\sigma_x(t)}{dt}\right] \\ &= -\frac{1}{N}\sum_{x=1}^{N} \text{Tr}\left[H_x \frac{d\sigma_x(t)}{dt}\right]\end{aligned} \tag{7}$$

where $H_x = \rho_x(\sigma_x^{\text{out}}(t))^{-1}$. This derivative quantifies how infinitesimal changes in the QNN unitaries influence the relative entropy cost, forming the basis for gradient-based parameter updates.

Training begins at $t = 0$, with all perceptron unitaries $W_p^l(0)$ randomly initialized. For each training pair $(\sigma_x^{\text{in}}, \rho_x)$, the input state σ_x^{in} is propagated through the network following the CP transition in (1), yielding the predicted output σ_x^{out}. At each layer, the perceptron unitaries act jointly on qubits from the preceding layer and on ancillary qubits initialized in the computational basis state $|0\rangle$. These ancillary qubits serve as auxiliary computational nodes that enhance the representational capacity of the network, enabling each perceptron to realize more general quantum transformations. The joint action of the perceptron unitary on its associated qubits and ancillas produces an intermediate mixed state, which is partially traced over the ancillas to yield the effective layer output. After each forward pass, the perceptron unitaries are updated according to

$$W_p^l(t + \delta) = e^{i\delta V_p^l(t)} W_p^l(t) \tag{8}$$

thereby preserving unitarity and physical consistency.

Proposition 1: The analytical gradient of the cost function is evaluated as

$$\frac{dC(t)}{dt} = -\frac{i}{N}\sum_{x=1}^{N}\sum_{l=1}^{\text{out}}\sum_{p=1}^{k_l} \text{Tr}\left(Q_p^l(t)\, V_p^l(t)\right) \tag{9}$$

where $Q_p^l(t)$ captures the contribution of the pth perceptron in the lth layer to the overall gradient, and is defined as

$$Q_p^l = \Bigg[\prod_{m=p}^{1} W_m^l (D^{l-1,l}) \prod_{m=1}^{p} {W_m^l}^\dagger, \prod_{m=p+1}^{k_l} {W_m^l}^\dagger \left(\mathbb{I}_l \otimes B^l\right) \prod_{m=k_l}^{p+1} W_m^l \Bigg] \tag{10}$$

with the forward-propagated and backward-propagated operators given by

$$D_x^{l-1,l} = D_x^{l-1} \otimes |0\cdots0\rangle\langle 0\cdots 0|_l$$
$$D_x^{l-1} = \left(\mathcal{T}^{l-1}\left(\ldots\left(\mathcal{T}^2\left(\mathcal{T}^1\left(\sigma_x^{\text{in}}\right)\right)\right)\ldots\right)\right) \tag{11}$$

and

$$B^l = \mathcal{G}^{l+1}\cdots\left(\mathcal{G}^{\text{out}}(H_x)\cdots\right) \tag{12}$$

where $\mathcal{G}^l = \sum_i M_i^\dagger X M_i$ is the adjoint CP channel corresponding to the forward map $\mathcal{T}^l(X) = \sum_i M_i X M_i^\dagger$.

Proof: The gradient is obtained by considering a perturbative evolution over a small increment δ, which gives the output state

$$\sigma_x^{\text{out}}(t+\delta) = \text{Tr}_{\text{in, hidden}}\Bigg[\left(\prod_{l=\text{out}}^{1}\prod_{p=k_l}^{1} e^{i\delta V_p^l(t)} W_p^l(t)\right) \times \left(\sigma_x^{\text{in}`} \otimes |0\cdots0\rangle\langle 0\cdots0|_{\text{hidden,out}}\right) \times \left(\prod_{l=1}^{\text{out}}\prod_{p=1}^{k_l} W_p^{l\dagger}(t) e^{-i\delta V_p^l(t)}\right)\Bigg]. \tag{13}$$

From this first-order approximation, the analytic gradient in equation (9) is derived. The full step-by-step derivation is provided in Appendix A. The backpropagation procedure is carried out using the adjoint channels of each layer: while the forward channels propagate the input density operators σ_x^{in}, the adjoint channels use the density matrix H_x backward from the output layer to the input. This is also depicted through Fig. 2.

To reach the minimum of the cost function with respect to the parameters faster, we formulate a gradient-based optimization by minimizing $\frac{dC(t)}{dt}$. Since this function is linear in $V_p^l(t)$, its extrema lie at $\pm\infty$. To obtain a finite and physically meaningful solution, we introduce a Lagrange multiplier $\lambda \in \mathbb{R}$, which constrains the parameterization of $V_p^l(t)$. Accordingly, each $V_p^l(t)$ is expanded in a fixed operator basis (e.g., the Pauli basis), and the constrained minimization problem can be expressed as

$$\min_{V^l_{p,\alpha_1,\ldots,\beta}} \left(\frac{dC(t)}{dt} + \lambda \sum_{\alpha_i,\beta}\left(V^l_{p,\alpha_1,\ldots,\beta}(t)\right)^2\right)$$
$$\text{where} \quad V_p^l(t) = \sum_{\alpha_1,\alpha_2,\ldots,\alpha_{m_k-1},\beta} V^l_{p,\alpha_1,\ldots,\alpha_{m_k-1},\beta}(t)$$

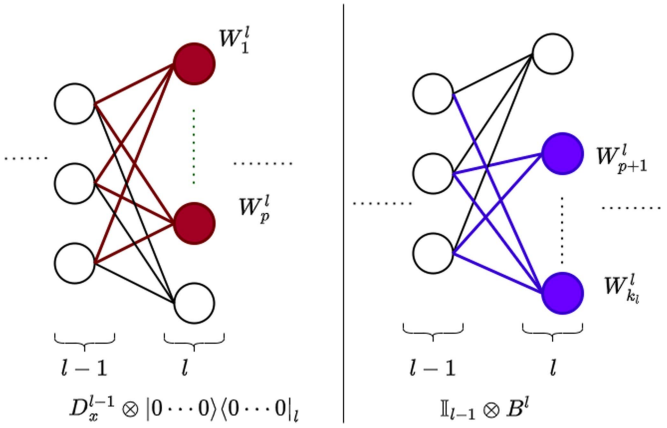


FIGURE 2. Illustration of the components of the parameter matrix in equation (15) for the *p*th perceptron in the *l*th layer.

$$\left(\Pi^{\alpha_1} \otimes \cdots \otimes \Pi^{\alpha_{m_k-1}} \otimes \Pi^{\beta}\right) \tag{14}$$

where the indices α_i of the basis coefficients $V^l_{p,\alpha_1,\ldots,\beta}(t)$ label the qubits from the previous layer, while β refers to the qubit in the current layer k. ■

Proposition 2: The parameter matrix corresponding to the lth layer and pth perceptron, denoted by V_p^l, is derived as

$$V_p^l = \frac{i\,2^{k_{l-1}}}{2N\lambda} \sum_x \text{Tr}_{\text{rest}}\left(Q_p^l(t)\right) \tag{15}$$

where Tr_{rest} denotes the partial trace over the complement of $\{\alpha_1, \ldots, \beta\}$.

Proof: This expression is obtained by analytically differentiating the optimization objective in equation (14) with respect to $V^l_{p,\alpha_1,\ldots,\beta}$. The complete derivation is provided in Appendix B. ■

With this formulation, the perceptrons for each layer are updated by tracing out the preceding layers and computing the corresponding commutation difference as defined in (10) and (15). The above propositions formalize the two complementary directions through which information propagates in the QNN. Proposition 1 ensures that each layer implements a well-defined CP evolution, so the intermediate states $D_x^0, D_x^1, \ldots, D_x^{\text{out}}$ represent a physically valid forward computation. Conversely, Proposition 1 shows that the sensitivity of the cost with respect to variations of the perceptron matrices is transported backward through the network via the adjoint maps $\mathcal{G}^l$, which play a role directly analogous to error backpropagation in classical neural networks. Thus, the forward-propagated density operators describe how input information flows through the architecture, while the backward-propagated adjoint operators capture how output deviations influence earlier layers. Together, these results yield the closed-form gradient expression of equation (9), enabling efficient analytical computation of the updates to the parameter matrices $V_p^l(t)$. Overall, the proposed QNN algorithm performs gradient updates through a combination of quantum feedforward propagation and adjoint channel

Algorithm 1: Relative-Entropy-Based QNN Training.

Require: Training pairs $\{(\sigma_x^{\text{in}}, \rho_x)\}$, initial perceptrons $W_p^l(0)$, step size δ.

$t \leftarrow 0$.
while stopping criterion not met **do**
  Forward pass:
  for each sample x **do**
    $\sigma_x^0 = \sigma_x^{\text{in}}$.
    for each layer $l = 1, \ldots, L$ **do**
      $D_x^{l-1,l} = \sigma_x^{l-1} \otimes |0\cdots0\rangle\langle 0\cdots0|_l$.
      $Y_x^l = W^l D_x^{l-1,l} W^{l\dagger}$.
      $\sigma_x^l = \text{Tr}_{l-1}(Y_x^l)$.
      Store σ_x^l.
    end for
    $\sigma_x^{\text{out}} = \sigma_x^L$.
  end for
  Backward seed preparation:
  for each sample x **do**
    $H_x = \rho_x(\sigma_x^{\text{out}})^{-1}$.
    $H_x \leftarrow (H_x + H_x^\dagger)/2$.
    Threshold negative eigenvalues; normalize $\text{Tr}(H_x) = 1$.
  end for
  Backward pass:
  for each sample x **do**
    $B_x^L = H_x$.
    for each $l = L-1, \ldots, 1$ **do**
      $B_x^l = \mathcal{G}^{l+1}(B_x^{l+1})$.
    end for
  end for
  Perceptron updates:
  for each layer $l = 1, \ldots, L$ **do**
    for each perceptron $p = 1, \ldots, k_l$ **do**
      Compute $Q_{x,p}^l$ for all samples using σ_x^{l-1} and B_x^l as equation (10).
      Aggregate $\{Q_{x,p}^l\}$ across samples to obtain Q_p^l.
      Form Hermitian $V_p^l(t)$.
      Update $W_p^l(t+\delta) = e^{i\delta V_p^l(t)} W_p^l(t)$.
    end for
  end for
  Cost evaluation:
  Compute $S_x = S(\rho_x \| \sigma_x^{\text{out}})$.
  $C(t) = \frac{1}{N}\sum_x S_x$.
  $t \leftarrow t + \delta$.
end while
Output: Trained perceptrons $W_p^l(t)$.

backpropagation, with the layer-wise information flow summarized in Fig. 3.

However, for the adjoint channel to be valid, a crucial requirement is that H_x must represent a valid density matrix, ensuring that the outcomes can be propagated through the adjoint channel in a physically realizable quantum domain.

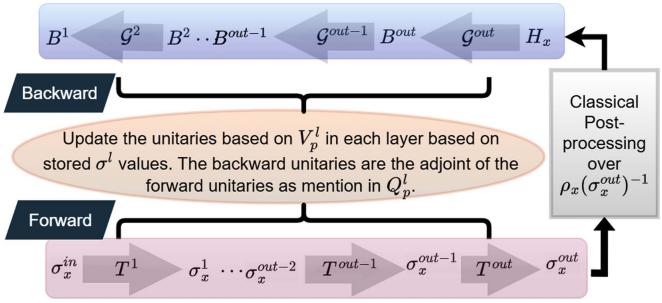


FIGURE 3. Flow diagram of the proposed QNN architecture, illustrating the layerwise forward channels $\mathcal{T}$ and the corresponding adjoint channels $\mathcal{G}$.

This guarantees that perceptrons W_p^l are updated consistently with the quantum state structure, allowing the QNN to handle mixed-state outputs while preserving the integrity of the relative-entropy-based cost function. However, the mathematical expression of H_x does not inherently satisfy the properties of a valid density matrix (Hermiticity, positivity, and unit trace). To enforce physical consistency, we implement a postprocessing correction by symmetrizing H_x as

$$H_x \leftarrow \frac{H_x + H_x^\dagger}{2} \tag{16}$$

which ensures Hermiticity. Subsequently, positivity and unit trace are enforced via eigenvalue thresholding. In this procedure, the eigenvalues of H_x are projected onto the probability simplex by setting any negative eigenvalues to zero and then renormalizing the remaining eigenvalues so that they sum to one [31]. This step is applied locally to the output-layer density matrix H_x, independent of the overall QNN architecture, making it generally applicable. Although such postprocessing introduces additional computational overhead, it significantly enhances numerical stability and ensures physically valid gradients. In contrast, fidelity-based optimization methods—which depend only on state overlaps—do not require this correction, but they lack the robustness of the relative-entropy-based formulation in mixed-state training scenarios. This iterative procedure is repeated until convergence, with the average relative entropy cost function in (5) being reevaluated and the time step incremented at each iteration.

Note: Our framework extends ideas from the (QIB) [25] while focusing on practical supervised quantum channel learning. Unlike QIB, which optimizes information-theoretic objectives involving mutual information between a reference system and a compressed representation, our framework directly minimizes the quantum relative entropy between predicted and target states. It also provides computationally efficient and numerically stable gradients, avoids the need for hyperparameter tuning, mitigates barren plateaus, and ensures physical consistency of the learned quantum channels.

C. PROPOSED POSTPROCESSING STEPS WITH COMPLEXITY ANALYSIS

H_x, derived from postprocessing $\rho_x(\sigma_x^{\text{out}})^{-1}$, serves as the basis for gradient-based optimization. The computation of $H_x = \rho_x(\sigma_x^{\text{out}})^{-1}$ involves inverting the output density matrix σ_x^{out}. Since this matrix is restricted to the output layer, its size depends only on the number of output qubits, not the full network width. In practice, we evaluate this inversion directly in our simulations, which is computationally feasible for moderate output dimensions. As an alternative, one may exploit the Woodbury–Sherman–Morrison identity, which updates the inverse efficiently when σ_x^{out} changes incrementally across training steps. In such cases, the cost can be reduced from a full $O(d^3)$ inversion to $O(d^2\,k + k^3)$, where $d = 2^{n_{\text{out}}}$ is determined by the output qubit number and k is the rank of the perturbation. Thus, while our implementation uses exact inversion, the Woodbury formulation provides a scalable option for larger output layers.

Moreover, as addressed above, to ensure that H_x constitutes a physically reliable density matrix, it is necessary to enforce Hermiticity, positivity, and unit-trace constraints through additional postprocessing steps, including symmetrization and eigenvalue thresholding. Computationally, symmetrization is inexpensive, requiring only $O(d^2) = O(2^{2n})$ arithmetic operations for a $d \times d$ matrix, where $d = 2^n$ is the Hilbert-space dimension of the output layer with n qubits. Even for larger output layers ($n > 10$), this step remains practical and does not constitute a computational bottleneck. Second, positivity and unit trace can be enforced via eigenvalue thresholding, which projects the eigenvalues of H_x onto the probability simplex by setting negative eigenvalues to zero and renormalizing [31]. While full eigendecomposition of a dense matrix scales as $O(d^3) = O(2^{3n})$, which grows with n, this encourages the consideration of scalable alternatives that maintain the physical validity of H_x while improving computational efficiency. These strategies provide practical pathways to extend the method to larger output layers without compromising the correctness of the adjoint channel. One potential approach is the diagonal perturbation method described in [32], in which a diagonal matrix $\Delta = \text{diag}(\delta_1, \ldots, \delta_d)$ is added to the reconstructed density matrix ρ_{out} to yield a corrected matrix $\rho_{\text{corr}} = \rho_{\text{out}} + \Delta$. This perturbation ensures strict diagonal dominance with nonnegative diagonal entries, thereby guaranteeing $\rho_{\text{corr}} \succeq 0$. The computational cost of checking diagonal dominance scales linearly as $O(d)$, making it highly efficient, particularly in scenarios where the off-diagonal elements of ρ_{out} are small relative to the diagonal.Moreover, when the network output σ_x^{out} or the target density matrix ρ_x exhibits an approximate low-rank structure—where most eigenvalues are negligible except for a few dominant ones—iterative methods, such as Lanczos [33] or Arnoldi [34], can be employed to compute the leading k eigenpairs efficiently. The computational cost of these iterative methods scales as $O(k\,d^2)$, which is significantly lower than full diagonalization for $k \ll d$, while still preserving accuracy.

In practice, the choice between exact and approximate strategies depends on the system size and the desired accuracy. Symmetrization is always feasible, full eigendecomposition is practical for small n, and diagonal perturbation or low-rank approximation approaches provide natural pathways to scale the method to larger output layers. While these alternative strategies were not implemented in the present simulations, they represent practical options for efficiently handling high-dimensional output layers if such scenarios arise. These proposals ensure that the adjoint channel can be applied and maintain compatibility of the QNN architecture with hybrid quantum-classical implementations, even when the output layer spans a moderately large Hilbert space ($d = 2^n$), where n is the number of qubits in the final layer.

III. RESULTS

We present simulation results for the proposed QNN architecture. The main objective is to evaluate its ability to learn unknown quantum channels that may correspond to either unitary transformations or more general CPTP maps, following the framework of Beer et al. [22]. Our simulations extend this task to mixed states using a relative entropy cost function, which represents realistic quantum conditions involving decoherence and statistical mixtures. Training on mixed-state data allows evaluation of the QNN's robustness and generalization capability in approximating an unknown quantum channel under practical noisy environments.

In each experiment, a single target quantum channel $\mathcal{T}$ is fixed, from which multiple datasets are generated. Random mixed quantum states X_i are constructed as convex combinations of k randomly generated pure states $|\phi_j\rangle$

$$\sigma_i^{in} = \frac{1}{\text{Tr}(\tilde{X}_i)} \sum_{j=1}^{k} w_j |\phi_j\rangle\langle\phi_j|, \quad w_j \sim \text{U}[0, 1] \tag{17}$$

where U[0, 1] denotes a continuous uniform distribution on [0, 1], and $\tilde{X}_i = \sum_{j=1}^{p} w_j |\phi_j\rangle\langle\phi_j|$ ensures normalization. Each input state X_i is then propagated through the target channel to produce the output

$$\rho_x = \mathcal{T}(\sigma_x^{\text{in}}) \tag{18}$$

forming a training pair $(\sigma_x^{\text{in}}, \rho_x)$. Repeating this process N times yields the complete dataset for the given channel.

The QNN is trained to approximate $\mathcal{T}$ using two cost formulations, one based on the existing fidelity loss [22] and the other on the proposed relative entropy loss. Within each experiment, k-fold cross-validation is used to monitor stability, tune hyperparameters, and obtain reliable performance estimates. A separate unseen test set is used to evaluate generalization on new input states. The entire procedure is repeated for several randomly generated channels, and performance metrics, such as training and test fidelities, relative entropy costs, and CIs are averaged across folds and experiments. This ensures that the reported results capture consistent QNN learning behavior across diverse channel realizations rather

than being specific to a single instance. Despite the limitations of classical simulation, the results show consistent convergence of all performance metrics, validating both the training algorithm and the proposed QNN architecture. The relative entropy–based cost provides improved stability and generalization compared with the fidelity-based baseline.

A. TRAINING PARAMETERS

The simulation parameters are summarized as follows. Multiple QNN configurations were tested to study scalability and expressive capacity, including baseline architectures [2,3,3,2] and [3,4,3] from Beer et al. [22], as well as deeper variants, such as [2,3,4,3,2] and [3,4,4,3]. Each configuration specifies the number of qubits in the input, hidden, and output layers, where the first and last entries correspond to the input and output qubits. For each randomly generated quantum channel, $N = 100$ input–output state pairs are generated as described in (17) and (18). The dataset is divided into training and test sets with a training ratio of 0.4, leaving 60% of the states as unseen samples for evaluating generalization.

Each experiment is repeated $N_{\text{exp}} = 10$ times to capture variability across different random channel realizations. Within each experiment, $k = 4$–fold cross-validation is used. Training proceeds for $T_{\max}$ rounds, and performance metrics are recorded every ten rounds to analyze convergence. In our implementation, $\lambda = 3$ and the step size is fixed at $\delta = 0.06$, chosen to ensure stable convergence, especially for the relative entropy cost, which may exhibit sharper gradients. The iteration variable t increases by δ at each round (for example, $t = 0.6$ after ten rounds and $t = 6$ after 100 rounds). These parameters are used consistently in both the fidelity-based and the relative-entropy–based training for fair comparison.

All simulations are performed using the QuTiP package [35], which ensures that all quantum operations, unitary evolutions, density matrices, and cost evaluations strictly follow quantum mechanical postulates. Statistical reliability is quantified across all Ξ independent experiments. For a given metric value $x_{\xi,\nu}^{(\kappa)}$ at training step ν in fold κ of experiment ξ, the mean and standard deviation are computed as

$$\bar{x}_\nu = \frac{1}{\Xi}\sum_{\xi=1}^{\Xi}\frac{1}{K}\sum_{\kappa=1}^{K} x_{\xi,\nu}^{(\kappa)} \tag{19}$$

$$s_\nu = \sqrt{\frac{1}{\Xi-1}\sum\nolimits_{\xi=1}^{\Xi}\left(\frac{1}{K}\sum_{\kappa=1}^{K} x_{\xi,\nu}^{(\kappa)} - \bar{x}_\nu\right)^2}. \tag{20}$$

The 95% CI at step ν is

$$\text{Ci}_\nu = t_{\text{crit},\,\Xi-1}\cdot\frac{s_\nu}{\sqrt{\Xi}} \tag{21}$$

where $t_{\text{crit},\,\Xi-1}$ is the Student's t-critical value for confidence level $1-\alpha$. These CIs provide a statistically grounded estimate of uncertainty in the mean metric, reflecting both random initialization and variability across folds and channel realizations. Plotting the mean with its CI offers a rigorous visualization of the QNN performance and expected fluctuations across independent runs.

B. RESULTS AND ANALYSIS ACROSS CONFIGURATIONS

To evaluate the scalability and representational capacity of the proposed QNN framework, we conduct experiments on three configurations corresponding to different input–output dimensions. These represent single qubit, two qubit, and three qubit channel learning tasks. In each case, the objective is to approximate an unknown quantum channel $\mathcal{E}$ from a finite set of mixed state input–output samples using either the fidelity based or relative entropy-based training objective. The following sections describe the observed training behavior, convergence, and generalization for each setting.

1) SINGLE-QUBIT CHANNEL LEARNING FOR DIFFERENT CONFIGURATIONS

This configuration represents the simplest case with single qubit input and output density matrices. To study the effect of network topology on learning, we tested QNNs with architectures [1,1,1], [1,3,1], and [1,2,3,2,1] using $T_{\max} = 150$. The [1,1,1] network acts as a minimal baseline, while deeper and wider networks add hidden perceptrons to improve expressivity. Fig. 4 shows the test outcomes where Fig. 4(a) compares the fidelity metric and Fig. 4(b) shows the relative entropy metric. Across all architectures, the proposed relative entropy-based method consistently surpasses the fidelity based baseline. The [1,3,1] model provides the greatest improvement, suggesting that moderate widening of hidden layers contributes more to learning efficiency than additional depth.

2) TWO-QUBIT CHANNEL LEARNING FOR DIFFERENT CONFIGURATIONS

The two qubit configuration increases the Hilbert space dimension to 4×4, expanding the parameter space and introducing richer entangling dynamics. To examine how network topology influences learning, we evaluated QNNs with architectures [2,2,2], [2,4,2], [2,3,3,2], and [2,3,4,3,2] using $T_{\max} = 350$. The [2,2,2] model, having limited hidden width, shows restricted capacity and lower performance. The [2,4,2] architecture achieves the highest improvement, indicating that increasing hidden layer width is more beneficial than increasing depth. Deeper models, such as [2,3,3,2] and [2,3,4,3,2], do not yield additional gains. Across all cases, the proposed relative entropy-based approach consistently outperforms the fidelity-based method in both metrics, as shown in Fig. 5. These observations confirm that wider hidden layers capture two qubit channel dynamics more effectively than simply deeper ones.

3) THREE-QUBIT CHANNEL LEARNING FOR DIFFERENT CONFIGURATIONS

The three-qubit configurations extends the Hilbert-space dimension to 8×8, significantly increasing the parameter

IEEE Transactions on Quantum Engineering

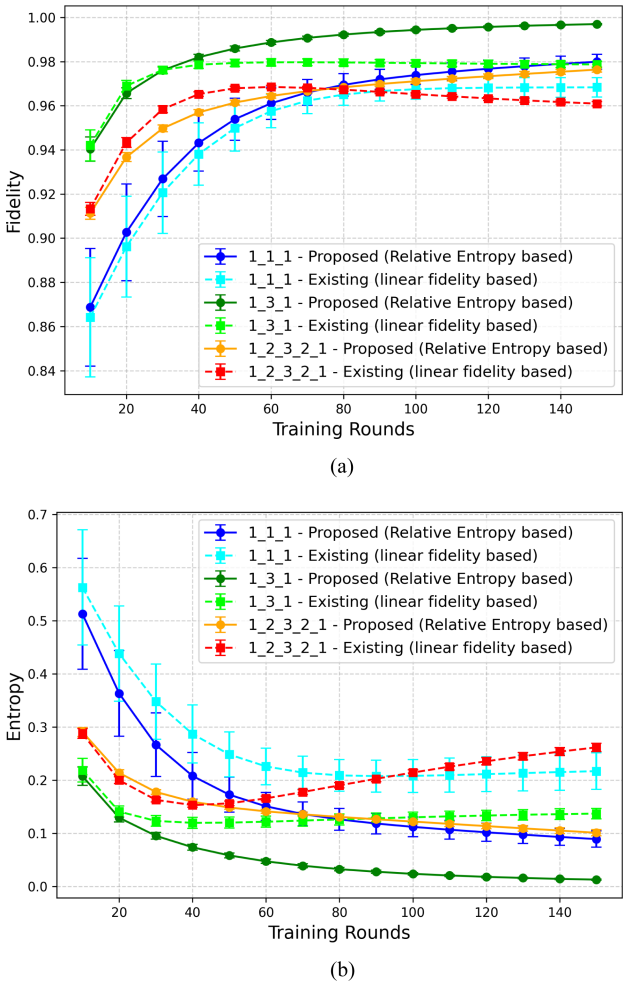


FIGURE 4. Test performance of different single-qubit QNN configurations. (a) Fidelity metric. (b) Relative Entropy metric. (a) Fidelity metric comparison for test dataset. (b) Relative Entropy metric comparison for test dataset.

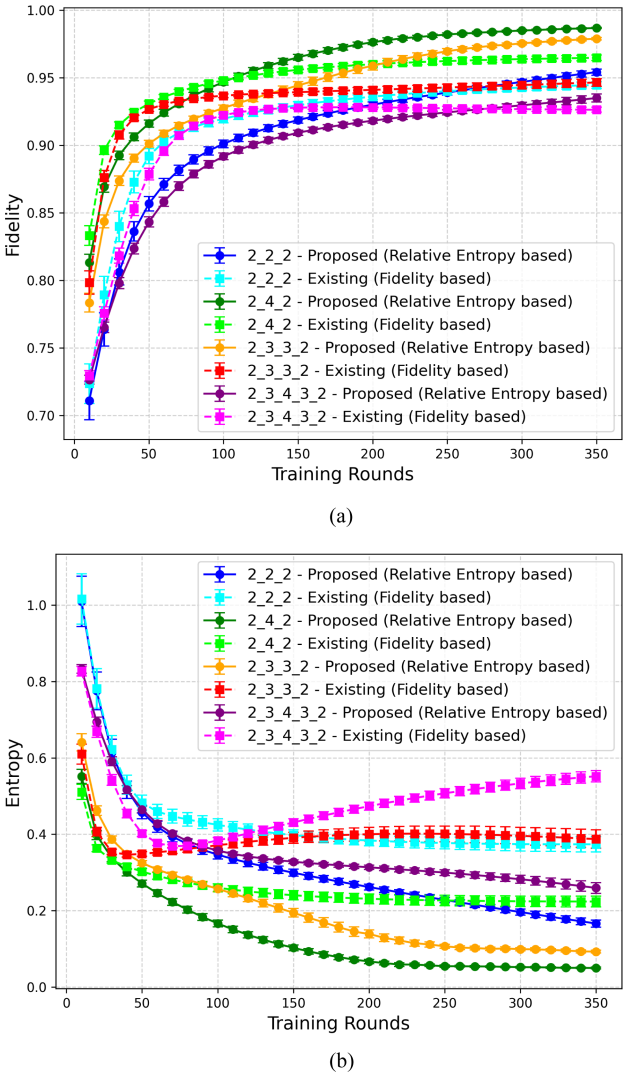


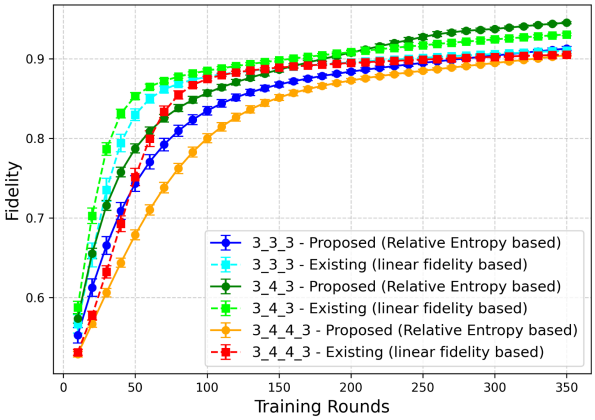
FIGURE 5. Test performance of different two-qubit QNN configurations. (a) Fidelity metric. (b) Relative Entropy metric. (a) Fidelity metric comparison for test dataset. (b) Relative Entropy metric comparison for test dataset.

space and introducing more complex entangling dynamics. To examine the influence of network depth and width on learning performance in this larger space, we evaluated multiple QNN configurations, including [3,3,3], [3,4,3], and [3,4,4,3] with $T_{\max} = 350$. The symmetric [3,3,3] network, with minimal hidden-layer width, serves as a baseline and exhibits limited expressive capacity. Increasing the width to [3,4,3] yields the most pronounced performance improvement, while deeper configurations, such as [3,4,4,3], do not provide significant additional gains. This observation indicates that, for three-qubit QNNs, expanding the hidden-layer width is more effective than simply adding depth. Across all configurations, the proposed relative-entropy–based method consistently outperforms the fidelity-based baseline in both fidelity and relative-entropy metrics, as illustrated in Fig. 6. These results further validate that layer width is a critical factor in capturing complex multiqubit quantum channel dynamics.

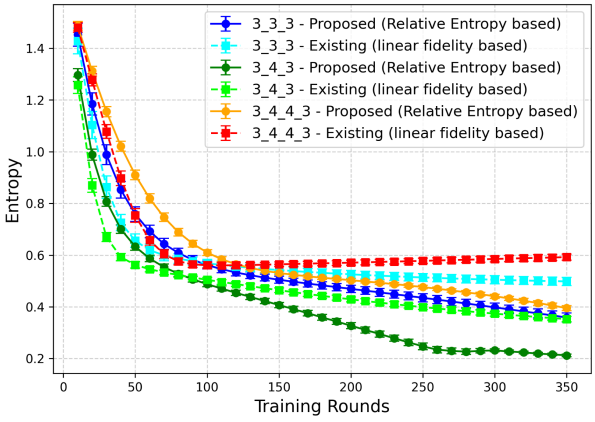
TABLE 1. Test Performance Comparison Showing Mean and CI for Entropy- and Fidelity-Based Training Across Different QNN Configurations

Configurations	Entropy		Fidelity		Advantage.
	Mean	CI	Mean	CI	(%)
[1, 1, 1]	0.08895	0.01536	0.21685	0.03459	59.00
[1, 3, 1]	0.01260	0.00130	0.13695	0.00993	90.81
[1, 2, 3, 2, 1]	0.10084	0.00571	0.26153	0.00797	61.43
[2, 2, 2]	0.16609	0.00928	0.37372	0.02035	55.56
[2, 4, 2]	0.04971	0.00229	0.22369	0.01390	77.77
[2, 3, 3, 2]	0.09254	0.00380	0.38723	0.02398	76.09
[2, 3, 4, 3, 2]	0.25927	0.01418	0.55178	0.01523	53.00
[3, 3, 3]	0.35879	0.01744	0.49834	0.01560	28.04
[3, 4, 3]	0.21232	0.00467	0.35271	0.00966	39.79
[3, 4, 4, 3]	0.39518	0.00980	0.59217	0.01233	33.28

The last column reports the relative advantage (%) of the entropy-based approach.



(a)



(b)

FIGURE 6. Test performance of different three-qubit QNN configurations. (a) Fidelity metric. (b) Relative Entropy metric. (a) Fidelity metric comparison for test dataset. (b) Relative Entropy metric comparison for test dataset.

Table 1 presents the final-step test-set performance of all QNN configurations for both training methods based on the relative-entropy and fidelity cost functions. The columns are organized by metric groups: "Entropy" and "Fidelity," each subdivided into the corresponding mean and 95% CI, followed by the relative advantage of the entropy-based method. Across all configurations, the relative-entropy–based training consistently achieves lower entropy values than the fidelity-based baseline, indicating a stronger alignment between the learned and target quantum channels. Shallow yet wider configurations, such as [1,3,1], [2,4,2], and [3,4,3], demonstrate the greatest relative advantages, suggesting that increasing layer width enhances learning efficiency more effectively than increasing depth. Deeper networks, while exhibiting slightly larger CIs due to fold-to-fold and experiment-to-experiment variability, still outperform their fidelity-based counterparts, as reflected in the CI analysis discussed in Section III-A. Overall, these results confirm that the proposed entropy-based training provides a statistically robust and practical alternative to fidelity-driven optimization.

C. IMPACT OF QNN ARCHITECTURE

The performance trend observed in our experiments aligns closely with the theoretical insights reported by Beer et al. [22], who demonstrated that the expressive capacity of a quantum perceptron network is primarily governed by the width of each layer rather than the depth of the overall architecture. The Figs. 4(b), 5(b), and 6(b) clearly indicate that a shallow architecture with adequate width proves more effective than simply stacking additional layers without meaningful gain. Consistent with this principle, our results indicate that configurations, such as [2,4,2] and [3,4,3], featuring a single, sufficiently wide hidden layer achieve superior convergence and generalization relative to deeper configurations like [2,3,4,3,2] and [3,4,4,3], despite the latter performing additional sequential transformations. The extra depth in these deeper models increases gradient diffusion and errors during postprocessing of unitary updates, which can slow or destabilize convergence. In contrast, the wider-but-shallow networks concentrate the expressive capacity within a single hidden layer, allowing them to model complex quantum channel correlations effectively while keeping the overall circuit depth minimal, making them well-suited for NISQ implementation.

When the hidden-layer width is kept fixed, as in configurations [2,4,2] and [3,4,3], we observe that the configurations with a smaller input–output dimension [2,4,2] achieves a higher relative-entropy advantage. This effect arises because a smaller input–output dimension generates output states with fewer eigenvalues, concentrating the entropy gradient over a compact eigenspectrum. In contrast, increasing the input–output dimension from [2,4,2] to [3,4,3] spreads the divergence signal across a larger spectral space. This dispersion, compounded by stronger CP-map projection effects in higher dimensional channels, slightly reduces the effective optimization gain per update. Despite this, the entropy-based formulation consistently outperforms fidelity-based training across both configurations, indicating that the proposed cost retains its corrective strength even as dimensionality grows.

D. DISCUSSION

The baseline training strategy updates the perceptron parameters using an analytic gradient derived from the linear fidelity cost function in (3), which measures state overlap via a trace inner product. Although effective for pure states, this approach performs suboptimally in mixed-state scenarios since it fails to capture the complete channel dynamics. As shown in our simulations, linear fidelity serves only as a proxy for the true Uhlmann fidelity in (4), which involves nonlinear matrix operations, such as square roots and provides a more accurate measure of distinguishability, between mixed quantum states.

In contrast, the proposed training scheme employs quantum relative entropy as defined in (5). This cost imposes stronger constraints on both eigenvector and eigenvalue alignment, offering a more comprehensive divergence measure than linear fidelity. Despite not directly optimizing

Uhlmann fidelity, the entropy-based training consistently achieves equal or higher Uhlmann fidelity in all experiments. This demonstrates that minimizing an information-theoretic divergence naturally enhances Uhlmann fidelity and improves generalization.

A detailed comparison reveals a distinct difference in convergence behavior. With the linear fidelity objective, the relative entropy may increase during intermediate rounds, reflecting that its gradient promotes local overlap without regulating spectral distribution. Consequently, the optimizer may yield states with high overlap but greater statistical distinguishability from the target, as observed in Fig. 6(b). In contrast, the entropy-based approach enforces structured learning by penalizing both eigenvector misalignment and spectral deviation, contracting the learned channel toward the target along an information-geometric path that preserves full state structure. This yields a smooth and monotonic entropy decay accompanied by stable fidelity growth, confirming that relative entropy provides a more principled and stable convergence trajectory than linear fidelity.

E. ROBUSTNESS TO DEPOLARIZATION NOISE

To further test the robustness of the proposed QNN framework, we evaluate its performance under depolarization noise [36], which serves as a representative model for general decoherence processes in realistic quantum systems. Depolarization effectively captures the loss of quantum coherence due to uncontrolled environmental interactions by uniformly randomizing the quantum state across all basis directions [37], [38]. In this sense, it acts as a generalized abstraction of more specific decoherence channels such as amplitude damping (energy relaxation) and phase damping (pure dephasing) [39], [40]. While these latter processes describe directional or basis-dependent decoherence mechanisms, the depolarizing channel encompasses their combined effect through an isotropic contraction of the Bloch sphere, thereby providing a comprehensive test of the network's stability under generic noise conditions [37].

Mathematically, the input states ρ^{in} are transformed into noisy states as

$$\rho'^{\text{in}} = (1-\zeta)\rho^{\text{in}} + \zeta\frac{I}{\dim(\rho)} \tag{22}$$

where ζ denotes the noise strength, ρ^{in} is the original input density matrix, and I is the identity operator of matching dimension. This operation models the depolarizing effect that drives the state progressively toward a maximally mixed state as ζ increases, thereby quantifying the degree of coherence loss. Within our framework, this noisy transformation can be viewed as a CPTP map, consistent with the mixed-state formalism used throughout our QNN design. Consequently, the same learning dynamics—based on the relative entropy cost function—remain valid even when the inputs are no longer pure. This property demonstrates a key advantage of our approach: it seamlessly accommodates general quantum noise models without modifying the optimization rule, since noise channels are naturally represented as CPTP maps acting on the input or intermediate states.

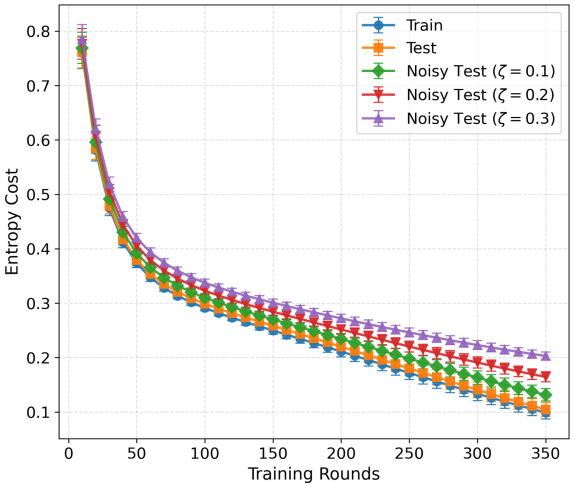


FIGURE 7. Relative entropy comparison of test data with various noise levels (ζ) for [2,3,3,2] network.

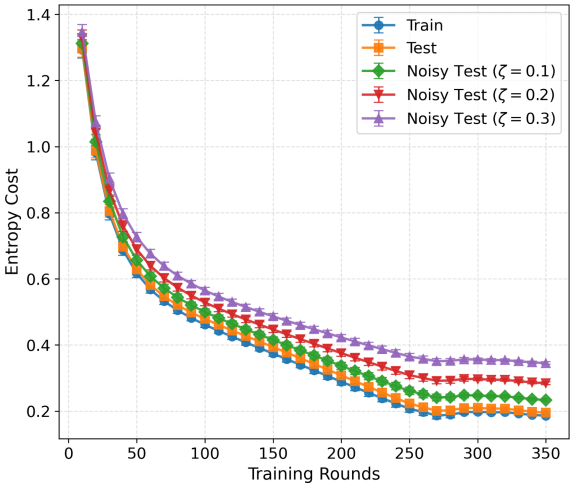


FIGURE 8. Relative entropy comparison of test data with various noise levels (ζ) for [3,4,3] network.

Figs. 7 and 8 present the performance of the [2,3,3,2] and [3, 4, 3] QNN architecture, where the train and clean test datasets remain identical, and additional noisy variants are generated by applying depolarization only to the test-state pairs for $\zeta \in [0.1, 0.2, 0.3]$. Despite increasing decoherence in these noisy test evaluations, the entropy-based cost function maintains convergence behavior comparable to the noise-free case. The final step values with CI are depicted through Table 2. Higher noise levels induce a gradual loss in reconstruction fidelity, yet the overall degradation remains controlled. These observations indicate that the proposed QNN framework preserves predictive stability under low to moderate depolarizing noise, reflecting suitability for NISQ-era conditions where decoherence is unavoidable.

TABLE 2. (Mean ± CI) for Different Configurations and Noise Levels

Metric	3-4-3	2-3-3-2
Train	0.1861 ± 0.004594	0.0989 ± 0.01183
Test	0.1951 ± 0.00529	0.1046 ± 0.01292
Noisy Test ($\zeta = 0.1$)	0.2334 ± 0.00455	0.1315 ± 0.01126
Noisy Test ($\zeta = 0.2$)	0.2843 ± 0.00407	0.1646 ± 0.00921
Noist Test ($\zeta = 0.3$)	0.3452 ± 0.00377	0.2030 ± 0.00696

IV. CONCLUSION AND FUTURE WORK

We introduced a relative-entropy–based cost function for training QNNs in mixed-state scenarios, providing a more informative and theoretically grounded metric than standard linear-fidelity objectives. By integrating this cost into existing QNN configurations through a modified backpropagation and postprocessing scheme, the framework efficiently learns CP maps while capturing the spectral structure of quantum channels. Although the additional postprocessing increases computational overhead, it significantly improves accuracy over linear fidelity-based baselines. These results confirm that relative entropy serves as a superior cost function for mixed-state QNN learning, offering richer information, smoother convergence, and a more physically meaningful measure of channel approximation. The robustness of this approach is further validated by evaluating performance under depolarizing noise, which serves as a representative model of generic decoherence processes in realistic quantum systems.

Future work will focus on extending this analysis to more physically realistic noise models, including amplitude damping, phase damping, and correlated or non-Markovian noise channels. Such studies will provide deeper insight into the generalization capabilities of QNNs under diverse quantum noise environments. In addition, mitigating hardware-centric implementation challenges and developing hardware-efficient entropy-based cost functions will be essential to reduce computational complexity. Exploring unsupervised or semi-supervised learning approaches could further allow QNNs to capture or characterize various quantum properties without the need for explicitly labeled outputs for every input. Overall, this framework extends the foundation for relative entropy based QNN-based learning of quantum channels, enabling robust modeling of noisy, and advancing QML for realistic quantum data.

APPENDIX

A. DERIVATION OF ANALYTIC GRADIENT

The unitaries $W_p^l(0)$ represent the initial perceptron operations, and each infinitesimal $e^{i\delta V_p^l(t)}$ updates the perceptrons layer by layer and perceptron by perceptron. The last layer $l = \text{out}$ corresponds to k_{out} number of perceptrons, $W_{k_{\text{out}}}^{\text{out}}$ is the last perceptron, and $V_{k_{\text{out}}}^{\text{out}}$ is its associated parameter matrix. The commutator $[V_p^l(t), \cdot]$ captures the first-order effect of each local parameter update, while the conjugated chain of subsequent unitaries forms the adjoint channel. Higher order terms $\mathcal{O}(\delta^2)$ are neglected. From the first order expansion of $\sigma_x(t+\delta)$ in (23), shown at the bottom of this page, we approximate $\dfrac{d\sigma_x(t)}{dt}$ as follows:

$$\begin{aligned}\frac{d\sigma_x(t)}{dt} &= \lim_{\delta\to 0}\frac{\sigma_x^{\text{out}}(t+\delta)-\sigma_x^{\text{out}}(t)}{\delta}\\ &= i\,\text{Tr}_{\text{in, hidden}}\Bigg[\sum_{l=1}^{\text{out}}\sum_{p=1}^{k_l}\Bigg(\Bigg(\prod_{l'=\text{out}}^{l}\prod_{p'=k_{l'}}^{p+1}W_{p'}^{l'}(t)\Bigg)\\ &\quad\times V_p^l(t)\prod_{l''=l}^{1}\prod_{p''=p}^{1}W_{p''}^{l''}(t)\left(\sigma_x^{\text{in}}\otimes|0\cdots0\rangle\langle 0\cdots0|\right)\\ &\quad\times\Bigg(\prod_{l''=1}^{l}\prod_{p''=1}^{p}W_{p''}^{l''\dagger}(t)\Bigg)\Bigg(\prod_{l'=l}^{\text{out}}\prod_{p'=p+1}^{k_{l'}}W_{p'}^{l'\dagger}(t)\Bigg)\Bigg)\Bigg]\end{aligned} \tag{24}$$

By substituting $\dfrac{d\sigma_x(t)}{dt}$ from (24) into (7), we calculate the cost-function gradient in (25), shown at the bottom of the next page (noting that out is used to denote the output layer). The second line applies the identity $\text{Tr}\,(A[V,X]) = \text{Tr}\,([X,A]\,V)$, where $A = \mathbb{I}_{\text{in+hidden}}\otimes H_x$, which is used for each individual perceptron. Using this relation, we define Q_p^l as shown in (26) at the bottom of the next page. In this definition, the first term (before the comma) corresponds to the feedforward chain, which propagates the input state from the initial perceptron W_1^1 in the first layer up to the current

$$\begin{aligned}\sigma_x^{\text{out}}(t+\delta) &= \text{Tr}_{\text{in, hidden}}\Bigg[\Bigg(\prod_{l=\text{out}}^{1}\prod_{p=k_l}^{1}e^{i\delta V_p^l(t)}W_p^l(t)\Bigg)\left(\sigma_x^{\text{in}}\otimes|0\cdots0\rangle\langle 0\cdots0|_{\text{hidden,out}}\right)\Bigg(\prod_{l=1}^{\text{out}}\prod_{p=1}^{k_l}W_p^{l\dagger}(t)e^{-i\delta V_p^l(t)}\Bigg)\Bigg]\\ &= \sigma_x^{\text{out}}(t)+i\delta\,\text{Tr}_{\text{in, hidden}}\Bigg[\sum_{l=1}^{\text{out}}\sum_{p=1}^{k_l}\Bigg(\Bigg(\prod_{l'=\text{out}}^{l}\prod_{p'=k_{l'}}^{p+1}W_{p'}^{l'}(t)\Bigg)\Bigg[V_p^l(t),\Bigg(\prod_{l''=l}^{1}\prod_{p''=p}^{1}W_{p''}^{l''}(t)\Bigg)\left(\sigma_x^{\text{in}}\otimes|0\cdots0\rangle\langle 0\cdots0|\right)\\ &\quad\times\Bigg(\prod_{l''=1}^{l}\prod_{p''=1}^{p}W_{p''}^{l''\dagger}(t)\Bigg)\Bigg]\Bigg(\prod_{l'=l}^{\text{out}}\prod_{p'=p+1}^{k_{l'}}W_{p'}^{l'\dagger}(t)\Bigg)\Bigg)\Bigg]+\mathcal{O}(\delta^2)\end{aligned} \tag{23}$$

IEEE Transactions on Quantum Engineering

perceptron W_p^l in layer l. The second term (after the comma) represents the remaining sequence of perceptrons extending to the output layer, conjugated and incorporating the operator $\mathbb{I}_{\text{in,hidden}} \otimes H_x$. This conjugated sequence implements the adjoint channel, capturing how the predicted output propagates backward through the network to influence the current perceptron. Together, this construction ensures that both the forward evolution leading to the perceptron and the backward influence from the output are fully accounted for when evaluating the impact of a local parameter update on the network output. By combining the definitions in (1), (11), and (12), (26) can be simplified, yielding the final expression for Q_p^l

$$Q_p^l = \Bigg[\prod_{m=p}^{1} W_m^l \left(D^{l-1,l}\right) \prod_{m=1}^{p} W_m^{l\,\dagger}, \prod_{m=p+1}^{k_l} W_m^{l\,\dagger} \left(\mathbb{I}_l \otimes B^l\right) \prod_{m=k_l}^{p+1} W_m^l \Bigg]. \tag{27}$$

B. OPTIMIZATION OF THE PARAMETER MATRIX

The optimization problem was first formulated as the minimization of the cost function in (14). To obtain the closed-form solution for the parameter matrices, we first expand (14) as shown in (28)

$$\begin{aligned}
&\min_{V^l_{p,\alpha_1,\ldots,\beta}} \left(\frac{dC(t)}{dt} + \lambda \sum_{\alpha_i,\beta} \left(V^l_{p,\alpha_1,\ldots,\beta}(t)\right)^2 \right) \\
&= \min_{V^l_{p,\alpha_1,\ldots,\beta}} \Bigg(-\frac{i}{N} \sum_x \mathrm{Tr}\left(Q^{\text{out}}_{k_{\text{out}}}(t) V^{\text{out}}_{k_{\text{out}}}(t) + \cdots \right. \\
&\quad \left. + Q^1_1(t) V^1_1(t)\right) + \lambda \sum_{\alpha_1,\ldots,\beta} \left(V^l_{p,\alpha_1,\ldots,\beta}(t)\right)^2 \Bigg) \\
&= \min_{V^l_{p,\alpha_1,\ldots,\beta}} \Bigg(-\frac{i}{N} \sum_x \mathrm{Tr}_{\alpha_1,\ldots,\beta} \left(\mathrm{Tr}_{\text{rest}} \left(Q^{\text{out}}_{k_{\text{out}}}(t) V^{\text{out}}_{k_{\text{out}}}(t) \right.\right. \\
&\quad \left.\left. + \cdots + Q^1_1(t) V^1_1(t)\right)\right) + \lambda \sum_{\alpha_1,\ldots,\beta} \left(V^l_{p,\alpha_1,\ldots,\beta}(t)\right)^2 \Bigg).
\end{aligned} \tag{28}$$

Here, the indices α_i label the qubits from the previous layer, while β refers to the qubit in the current layer k. The operator $\mathrm{Tr}_{\alpha_1,\ldots,\beta}(\cdot)$ denotes the partial trace taken over these qubits, and $\mathrm{Tr}_{\text{rest}}(\cdot)$ represents the trace over all other qubits that do not belong to this set. Next we take the derivative of the equation (28) with respect to the $V^l_{p,\alpha_1,\ldots,\beta}(t)$ and set the derivative to zero to minimize (28) and depicted as follows:

$$\begin{aligned}
V^l_{p,\alpha_1,\ldots,\beta}(t) = \frac{i}{2N\lambda} \sum_x \mathrm{Tr}_{\alpha_1,\ldots,\beta} \\
\left(\mathrm{Tr}_{\text{rest}} \left(Q^l_p(t)\right) \left(\Pi^{\alpha_1} \otimes \cdots \otimes \Pi^{\beta}\right)\right).
\end{aligned} \tag{29}$$

$$\begin{aligned}
\frac{dC(t)}{dt} &= -\frac{i}{N} \sum_x \mathrm{Tr}\Bigg(\mathrm{Tr}_{in,\text{hidden}} \Bigg(\mathbb{I}_{in,\text{hidden}} \otimes H_x \Big(\Big[V^{\text{out}}_{k_{\text{out}}}(t), W^{\text{out}}_{k_{\text{out}}}(t) \cdots W^1_1 (\sigma^{\text{in}}_x \otimes |0\cdots 0\rangle \\
&\quad \langle 0\cdots 0|_{\text{out,hidden}}) W_1^{1\dagger} \cdots W^{\text{out}\dagger}_{k_{\text{out}}}(t) \Big] + \cdots + W^{\text{out}}_{k_{\text{out}}}(t) \cdots W^2_1 \Big[V^1_1(t), W^1_1(t) \\
&\quad \left(\sigma^{\text{in}}_x \otimes |0\cdots 0\rangle\langle 0\cdots 0|_{\text{out,hidden}} W_1^{1\dagger}(t)\right) \Big] W_1^{2\dagger}(t) \cdots W^{\text{out}\dagger}_{k_{\text{out}}}(t) \Big) \Big) \Bigg) \\
&= -\frac{i}{N} \sum_x \mathrm{Tr}\Bigg(\Big[W^{\text{out}}_{k_{\text{out}}}(t) \cdots \left(\sigma^{\text{in}}_x \otimes |0\cdots 0\rangle\langle 0\cdots 0|\right)_{\text{hidden,out}} \cdots W^{\text{out}}_{k_{\text{out}}}(t)^\dagger, \mathbb{I}_{\text{in,hidden}} \otimes H_X \Big] V^{\text{out}}_{k_{\text{out}}}(t) \\
&\quad + \cdots + \Big[W^1_1(t) \left(\sigma^{\text{in}}_x \otimes |0\cdots 0\rangle\langle 0\cdots 0|\right)_{\text{hidden,out}} W^1_1(t)^\dagger, W^1_2(t)^\dagger \cdots W^{\text{out}}_{k_{\text{out}}}(t)^\dagger \\
&\quad (\mathbb{I}_{\text{in+hidden}} \otimes H_x) W^{\text{out}}_{k_{\text{out}}}(t) \cdots W^1_2(t) \Big] V^1_1(t) \Bigg) \\
&= -\frac{i}{N} \sum_x \mathrm{Tr}\left(Q^{\text{out}}_{k_{\text{out}}}(t) V^{\text{out}}_{k_{\text{out}}}(t) + \cdots + Q^1_1(t) V^1_1(t) \right) = -\frac{i}{N} \sum_{x=1}^{N} \sum_{l=1}^{\text{out}} \sum_{p=1}^{k_l} \mathrm{Tr}\left(Q^l_p(t)\, V^l_p(t) \right)
\end{aligned} \tag{25}$$

$$\begin{aligned}
Q^l_p(t) = \Big[&W^l_p(t) W^l_{p-1}(t) \cdots W^1_1(t) \left(\sigma^{\text{in}}_x \otimes |0\cdots 0\rangle_{\text{hidden,out}} \langle 0\cdots 0|\right) W_1^{1\dagger}(t) \cdots W^{l\dagger}_{p-1}(t) W^{l\dagger}_p(t), \\
&W^{l\dagger}_{p+1}(t) W^{l\dagger}_{p+2}(t) \cdots W^{\text{out}\dagger}_{k_{\text{out}}-1}(t) W^{\text{out}\dagger}_{k_{\text{out}}}(t) \left(I_{\text{in,hidden}} \otimes H_x\right) W^{\text{out}}_{k_{\text{out}}}(t) W^{\text{out}}_{k_{\text{out}}-1}(t) \cdots W^l_{p+2}(t) W^l_{p+1}(t) \Big]
\end{aligned} \tag{26}$$

Finally, the closed-form expression for V_p^l is given as follows:

$$V_p^l(t) = \frac{i}{2N\lambda} \sum_{\alpha_1,\ldots,\beta} \sum_x \mathrm{Tr}_{\alpha_1,\ldots,\beta} \left(\mathrm{Tr}_{\mathrm{rest}} \left(Q_p^l(t) \right) \right. \\ \left. \left(\Pi^{\alpha_1} \otimes \cdots \otimes \Pi^{\beta} \right) \right) \left(\Pi^{\alpha_1} \otimes \cdots \otimes \Pi^{\beta} \right) \\ = \frac{i 2^{k_{l-1}}}{2N\lambda} \sum_x \mathrm{Tr}_{\mathrm{rest}} \left(Q_p^l(t) \right). \quad (30)$$

REFERENCES

[1] J.-G. Liu and L. Wang, "Differentiable learning of quantum circuit born machines," *Phys. Rev. A*, vol. 98, no. 6, 2018, Art. no. 062324, doi: 10.1103/PhysRevA.98.062324.

[2] P.-L. Dallaire-Demers and N. Killoran, "Quantum generative adversarial networks," *Phys. Rev. A*, vol. 98, no. 1, 2018, Art. no. 012324, doi: 10.1103/PhysRevA.98.012324.

[3] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum Boltzmann machine," *Phys. Rev. X*, vol. 8, no. 2, 2018, Art. no. 021050, doi: 10.1103/PhysRevX.8.021050.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 25, pp. 84–90, 2012.

[5] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," *Phys. Rev. A*, vol. 101, no. 3, 2020, Art. no. 032308, doi: 10.1103/PhysRevA.101.032308.

[6] A. Macaluso, L. Clissa, S. Lodi, and C. Sartori, "A variational algorithm for quantum neural networks," in *Proc. 20th Int. Conf. Comput. Sci.*, 2020, pp. 591–604, doi: 10.1007/978-3-030-50433-5_45.

[7] Y. Li, R.-G. Zhou, R. Xu, J. Luo, and W. Hu, "A quantum deep convolutional neural network for image recognition," *Quantum Sci. Technol.*, vol. 5, no. 4, 2020, Art. no. 044003, doi: 10.1088/2058-9565/ab9f93.

[8] I. Cong, S. Choi, and M. D. Lukin, "Quantum convolutional neural networks," *Nature Phys.*, vol. 15, no. 12, pp. 1273–1278, 2019, doi: 10.1038/s41567-019-0648-8.

[9] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Phys. Rev. Lett.*, vol. 100, no. 16, 2008, Art. no. 160501, doi: 10.1103/PhysRevLett.100.160501.

[10] M. Henderson, S. Shakya, S. Pradhan, and T. Cook, "Quanvolutional neural networks: Powering image recognition with quantum circuits," *Quantum Mach. Intell.*, vol. 2, no. 1 2020, Art. no. 2, doi: 10.1007/s42484-020-00012-y.

[11] I. Kerenidis, J. Landman, and A. Prakash, "Quantum algorithms for deep convolutional neural networks," 2019, *arXiv:1911.01117*, doi: 110.48550/arXiv.1911.01117.

[12] M. Larocca, N. Ju, D. Garcıá-Martıń, P. J. Coles, and M. Cerezo, "Theory of overparametrization in quantum neural networks," *Nature Comput. Sci.*, vol. 3, no. 6, pp. 542–551, 2023, doi: 10.1038/s43588-023-00467-6.

[13] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, "The power of quantum neural networks," *Nature Comput. Sci.*, vol. 1, no. 6, pp. 403–409, 2021, doi: 10.1038/s43588-021-00084-1.

[14] Y. Du, M.-H. Hsieh, T. Liu, S. You, and D. Tao, "Learnability of quantum neural networks," *PRX Quantum*, vol. 2, no. 4, 2021, Art. no. 040337, doi: 10.1103/PRXQuantum.2.040337.

[15] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Commun.*, vol. 9, no. 1, 2018, Art. no. 4812, doi: 10.1038/s41467-018-07090-4.

[16] L. Schatzki, M. Larocca, Q. T. Nguyen, F. Sauvage, and M. Cerezo, "Theoretical guarantees for permutation-equivariant quantum neural networks," *npj Quantum Inf.*, vol. 10, no. 1, 2024, Art. no. 12, doi: 10.1038/s41534-024-00804-1.

[17] A. Arrasmith, M. Cerezo, P. Czarnik, L. Cincio, and P. J. Coles, "Effect of barren plateaus on gradient-free optimization," *Quantum*, vol. 5, 2021, Art. no. 558, doi: 10.22331/q-2021-10-05-558.

[18] M. Kashif and S. Al-Kuwari, "ResQNets: A residual approach for mitigating barren plateaus in quantum neural networks," *EPJ Quantum Technol.*, vol. 11, no. 1, 2024, Art. no. 4, doi: 10.1140/epjqt/s40507-023-00216-8.

[19] L. Friedrich and J. Maziero, "Avoiding barren plateaus with classical deep neural networks," *Phys. Rev. A*, vol. 106, no. 4, 2022, Art. no. 042433, doi: 10.1103/PhysRevA.106.042433.

[20] S. A. Wilkinson and M. J. Hartmann, "Evaluating the performance of sigmoid quantum perceptrons in quantum neural networks," 2022, *arXiv:2208.06198*, doi: 10.48550/arXiv.2208.06198.

[21] F. V. Massoli, L. Vadicamo, G. Amato, and F. Falchi, "A leap among quantum computing and quantum neural networks: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, 2022, doi: 10.1145/3529756.

[22] K. Beer et al., "Training deep quantum neural networks," *Nature Commun.*, vol. 11, no. 1, 2020, Art. no. 808, doi: 10.1038/s41467-020-14454-2.

[23] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," *Phys. Rev. A*, vol. 99, no. 3, 2019, Art. no. 032331, doi: 10.1103/PhysRevA.99.032331.

[24] X. Pan et al., "Deep quantum neural networks on a superconducting processor," *Nature Commun.*, vol. 14, no. 1, 2023, Art. no. 4006, doi: 10.1038/s41467-023-39785-8.

[25] A. B. Çatlı and N. Wiebe, "Training quantum neural networks using the quantum information bottleneck method," *J. Phys. A: Math. Theor.*, vol. 57, no. 37, 2024, Art. no. 375302, doi: 10.1088/1751-8121/ad6daf.

[26] F. Marquardt and A. Püttmann, "Introduction to dissipation and decoherence in quantum systems," 2008, *arXiv:0809.4403*, doi: 10.48550/arXiv.0809.4403.

[27] M. L. Olivera-Atencio, L. Lamata, and J. Casado-Pascual, "Benefits of open quantum systems for quantum machine learning," *Adv. Quantum Technol.*, to be published, 2023, Art. no. 2300247, doi: 10.1002/qute.202300247.

[28] N. A. Peters, T.-C. Wei, and P. G. Kwiat, "Mixed-state sensitivity of several quantum-information benchmarks," *Phys. Rev. At., Mol., Opt. Phys.*, vol. 70, no. 5, 2004, Art. no. 052309, doi: 10.1103/PhysRevA.70.052309.

[29] V. Vedral, "The role of relative entropy in quantum information theory," *Rev. Modern Phys.*, vol. 74, no. 1, 2002, Art. no. 197, doi: 10.1103/RevModPhys.74.197.

[30] Z. Goldfeld, D. Patel, S. Sreekumar, and M. M. Wilde, "Quantum neural estimation of entropies," *Phys. Rev. A*, vol. 109, no. 3, 2024, Art. no. 032431, doi: 10.1103/PhysRevA.109.032431.

[31] J. A. Smolin, J. M. Gambetta, and G. Smith, "Efficient method for computing the maximum-likelihood quantum state from measurements with additive Gaussian noise," *Phys. Rev. Lett.*, vol. 108, no. 7, 2012, Art. no. 070502, doi: 10.1103/PhysRevLett.108.070502.

[32] S. Mondal and A. K. Dutta, "A modified least squares-based tomography with density matrix perturbation and linear entropy consideration along with performance analysis," *New J. Phys.*, vol. 25, no. 8, 2023, Art. no. 083051, doi: 10.1088/1367-2630/acf187.

[33] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Standards*, vol. 45, no. 4, pp. 255–282, 1950, doi: 10.6028/jres.045.026.

[34] W. E. Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quart. Appl. Math.*, vol. 9, no. 1, pp. 17–29, 1951, doi: 10.1090/qam/42792.

[35] J. R. Johansson, P. D. Nation, and F. Nori, "QuTiP: An open-source Python framework for the dynamics of open quantum systems," *Comput. Phys. Commun.*, vol. 183, no. 8, pp. 1760–1772, 2012, doi: 10.1016/j.cpc.2012.02.021.

[36] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, vol. 2. Cambridge, U.K.: Cambridge Univ. Press, 2001, doi: 10.1017/CBO9780511976667.

[37] C. Cafaro and S. Mancini, "Characterizing the depolarizing quantum channel in terms of Riemannian geometry," *Int. J. Geometric Methods Modern Phys.*, vol. 9, no. 02, 2012, Art. no. 1260020, doi: 10.1142/S0219887812600201.

[38] L.-M. Yang, B. Chen, S.-M. Fei, and Z.-X. Wang, "Dynamics of coherence-induced state ordering under Markovian channels," *Front. Phys.*, vol. 13, no. 5, 2018, Art. no. 130310, doi: 10.1007/s11467-018-0780-4.

[39] K. F. Romero and R. L. Franco, "Simple non-Markovian microscopic models for the depolarizing channel of a single qubit," *Physica Scripta*, vol. 86, no. 6, 2012, Art. no. 065004, doi: 10.1088/0031-8949/86/06/065004.

[40] T. Ahmed, M. Kashif, A. Marchisio, and M. Shafique, "Quantum neural networks: A comparative analysis and noise robustness evaluation," *Sci. Res.*, vol. 15, 2025, Art. no. 33654, doi: 10.1038/s41598-025-17769-6.