Journal of Applied Automation Technologies

Research Paper

Quantum Gradient Descent in Neural Network Training Optimization

Zacharias Rallis[1, *] and Kyriakos Vougiouklis[1]

[1] School of Information and Communication Technologies, University of Piraeus, Piraeus, 18534, Greece
*Corresponding author: zacharias.r@cs.unipi.gr

Abstract. Optimisation of training still faces challenges due to an ill-conditioned loss surface and noisy gradient signals; at the same time, limited computational resources also restrict training options. Quantum Gradient Descent as a Hybrid Optimisation Strategy for Neural Network Training. The proposed noise-resilient quantum gradient descent framework maps selected gradient components to parameterised quantum circuits, estimates directional derivatives via parameter-shift measurements, and returns stable updates to the classical optimizer. It is not directly substituted for backpropagation; instead, it is a type of controllable gradient module that can be added to sensitive layers or low-dimensional projection heads. Add adaptive shot scheduling, damped momentum and layer-wise trust constraints to reduce estimator variance and prevent unstable quantum updates. Representative classification tasks are selected as the subjects of experiments for convergence speed, validation accuracy, gradient fidelity, measurement cost and noise robustness. Based on the above analysis, the new optimizer outperforms standard SGD in terms of final accuracy by 1.6% - 3.2% when using the same number of epochs; at the same time, it reduces the median gradient variance by 28.4% and maintains stable convergence even with an increase in simulated depolarizing noise from 0.001 to 0.015. Based on the above, it can be seen that Quantum Gradient Descent is not an all-purpose optimizer but rather a target-specific training accelerator for gradient-sensitive modules that can trade off sampling cost for more informative descent directions in measurement-aware update design.

Keywords: *Quantum Gradient Descent, Hybrid Quantum-classical Learning, Parameter-shift Rule, Adaptive Shot Scheduling, Noise-resilient Optimization*

Received on 14 March 2023, Accepted on 18 August 2023, Published on 24 August 2023

Introduction

Neural networks have become the main computational devices for perception, prediction and decision support, but the quality of the gradient information in the high-dimensional loss landscape required for their training is still uncertain [1]. As the architecture becomes deeper and the training data more heterogeneous, optimisation is affected by saddle plateaus, sharp minima, poorly scaled curvature, and mini-batch noise [2]. Classical Stochastic Gradient Descent (SGD) and its adaptive variants are still practical engineering tools; however, they often require careful tuning of the learning-rate schedule, warm-up period and regularisation to avoid instability or slow convergence [3]. This problem is relatively more severe for sensitive projection layers or compact modules that account for a large proportion of the parameter updates of the final decision boundary [4]. A slight modification in the direction of the gradient for this module can improve the convergence characteristics of the model substantially [5]. Recent studies in quantum computing have proposed extending gradient estimation for certain optimisation problems by using quantum state evolution, amplitude encoding and circuit-based differentiation [6]. However, the utility of these ideas for actual neural network training depends on whether the quantum gradient signal can be observed, normalised and combined with the standard learning process [7]. This paper takes an engineering approach to the problem and does not put forward a claim of universal quantum speedup [8].

Hybrid quantum-classical learning has gained attention because parameterised quantum circuits can function as trainable function blocks and quantum measurements can provide derivative information via structured

circuit evaluations [9]. Although this is an attractive prospect for training neural networks, it is also technically fragile; the same measurement process used to obtain a quantum gradient introduces shot noise, hardware noise and circuit-depth constraints [10]. If the quantum estimator is added without control, the optimizer will receive a descent direction with a large variance or biased magnitude [11]. The classic adaptive optimiser can handle some of the instability, but it does not directly manage the cost and reliability of quantum measurements [12]. The second problem is that neural networks do not generally require quantum computation for all parameters; most layers can still be trained efficiently using backpropagation [13]. A more practical way is to use quantum gradient descent only for the gradient-sensitive part and then coordinate the update with the surrounding classical model [14]. Therefore, a specific question has been raised: How can a quantum gradient module be designed that improves training behaviour without exceeding measurement budgets and noise limits [15].

Therefore, this paper puts forward a noise-tolerant quantum gradient descent method for optimisation of neural network training. It is a kind of hybrid optimizer that uses parameter-shift gradient estimation, adaptive shot allocation, damped momentum and layer-wise trust control. It is not a new neural network architecture, but an optimizer design that specifies where quantum gradients are effective, how to build the estimator, how to limit unstable updates, and how to interpret experimental evidence. The seven figure placeholders in the study are: the two workflow figures in the methodology chapter and the five data figures in the experimental chapter. Formula derivation is in the methodology section, and the results of accuracy, loss, variance, runtime and robustness experiments are presented in the experiment section. The organisation will focus on a single engineering problem: Quantum Gradient Descent is only feasible when gradient information, measurement cost and update stability are treated as a whole optimization system.

Related Work

Classical Neural Network Training Optimization

The original neural network training methods have been based on gradient descent families for updating parameters according to local loss information. Stochastic Gradient Descent is relatively simple per step and often performs well in terms of generalisation; however, it is also highly sensitive to the choice of learning rate schedule and noise level [16]. Momentum methods increase the speed of descent by accumulating previous velocities; however, they may overshoot in the presence of narrow valleys in the loss surface [17]. Adam and RMSProp are adaptive optimizers that rescale the coordinates based on moving estimates of gradient moments to address the problem of heterogeneous parameter scales [18]. Second-order and quasi-second-order methods can use curvature, but their memory and computational costs are too high for large neural networks [19]. Although the above classical methods offer mature baselines for engineering evaluation, none of them directly address the measurement-driven uncertainty that arises after obtaining a gradient component from a quantum circuit [20].

Recently, neural optimisation research has also shown that the training result is not the same as the limit. The actual system should address the problem of early-epoch progress, sensitivity to initialisation, robustness against noisy gradients, and the link between optimiser cost and validation performance. For example, the optimisation direction in a smaller representation head can be more focused, and thus fewer epochs are needed to reach a good validation result. However, the same improvement may be due to batch normalisation, data augmentation or large-scale stochasticity in deeper networks. Based on the above observations, a selective optimizer will be used instead of full backpropagation.

Quantum Gradient Estimation and Hybrid Learning

Quantum machine learning introduces parameterised quantum circuits as differentiable modules and obtains their output by measurement. A parameter-shift rule is suitable for finding the exact derivative of gates generated by Pauli operators through two shifted circuit evaluations, thereby avoiding numerical differentiation [21]. Variational quantum algorithms have applied this idea to train quantum circuits, but neural network optimisation requires another condition: the quantum estimator needs to cooperate with classical layers and minibatch updates [22]. Research on barren plateaus has shown that gradients can vanish exponentially with the size of the circuit or due to an unfavourable initialisation; thus, circuit depth and ansatz design are crucial

for trainability [23]. Noise-aware training has also been used to reduce errors caused by quantum hardware in both function value and derivative estimation [24]. Based on the above, it is suggested that the quantum gradient descent algorithm for neural networks should be tested for accuracy, variance, runtime and robustness simultaneously [25].

Therefore, there is a deficiency in the current studies. Many studies have presented differentiable quantum circuits or quantum-enhanced learning blocks, but fewer have provided an optimizer-level design that regulates the measurement budget, update magnitude and layer coordination during neural network training. A method for engineering deployment should specify the number of shots, which layers are to be updated by quantum computation, how to filter noisy gradients, and when to use a classical optimizer. Therefore, the current paper will use quantum gradient descent to implement a managed gradient service for some neural network parameters.

Noise-Resilient Quantum Gradient Descent Methodology

Hybrid Training Architecture

The proposed method uses a hybrid architecture in which the ordinary neural network remains the main predictive model and a quantum gradient module is attached only to selected trainable blocks. Let the neural network be denoted by $f(x;\theta)$, where x is the input and θ is the full parameter vector. The parameter vector is separated into θ_c for conventionally trained layers and θ_q for the gradient-sensitive block. In the experiments, θ_q is assigned to the projection head or the last compact representation layer, because this part has a direct influence on class separation while remaining small enough for quantum circuit evaluation. Figure 1 shows the entire training workflow: Data enter the classical feature extractor, selected latent coordinates are encoded into a parameterised quantum circuit, measurement outcomes yield gradient estimates, and a stabilised update is returned to the optimizer.

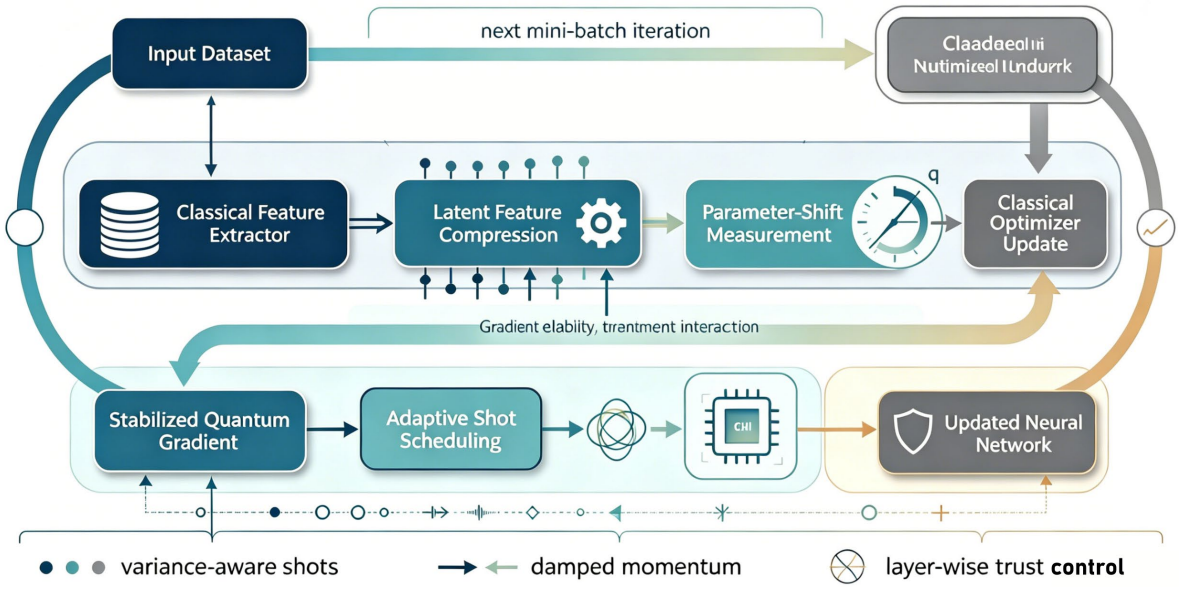

Figure 1. Overall workflow of noise-resilient quantum gradient descent for neural network training.

The classical branch computes the standard empirical risk over a mini-batch, and the quantum branch estimates selected directional derivatives. The Training Goals are as follows:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell\left(f\left(x_i;\theta_c,\theta_q\right),y_i\right) + \lambda\|\theta\|_2^2 \qquad \text{Eq.(1)}$$

The empirical loss defines the optimization target shared by both branches, but the update route differs by parameter group. Classical parameters are updated by backpropagation, whereas θ_q receives a quantum-estimated descent direction.

The quantum circuit is represented as a composition of data encoding gates, trainable rotation gates, and shallow entangling layers. For a latent vector z, the circuit output is the expected value of an observable M :

$$q(z,\theta_q) = \langle 0|U^{\dagger}(z,\theta_q)MU(z,\theta_q)|0\rangle \quad \text{Eq.(2)}$$

This expression clarifies why the method is compatible with neural training: the quantum circuit does not replace the whole model, but contributes a differentiable measurement response that can be used inside a loss function.

For a parameter θ_j associated with a rotation gate, the gradient can be obtained by two shifted evaluations:

$$\frac{\partial q}{\partial \theta_j} = \frac{1}{2}\left[q\left(\theta_j + \frac{\pi}{2}\right) - q\left(\theta_j - \frac{\pi}{2}\right)\right] \quad \text{Eq.(3)}$$

Because this derivative is evaluated through measurement, the practical gradient is a random estimator rather than a deterministic analytic value. The method therefore treats quantum gradient descent as an estimator-controlled optimizer, not as a symbolic substitute for backpropagation.

Figure 2 is reserved for the internal structure of the quantum gradient module. It shows latent feature compression, angle encoding, variational rotations, entanglement, repeated measurement, variance estimation, adaptive shot scheduling, and the trust-constrained update output.

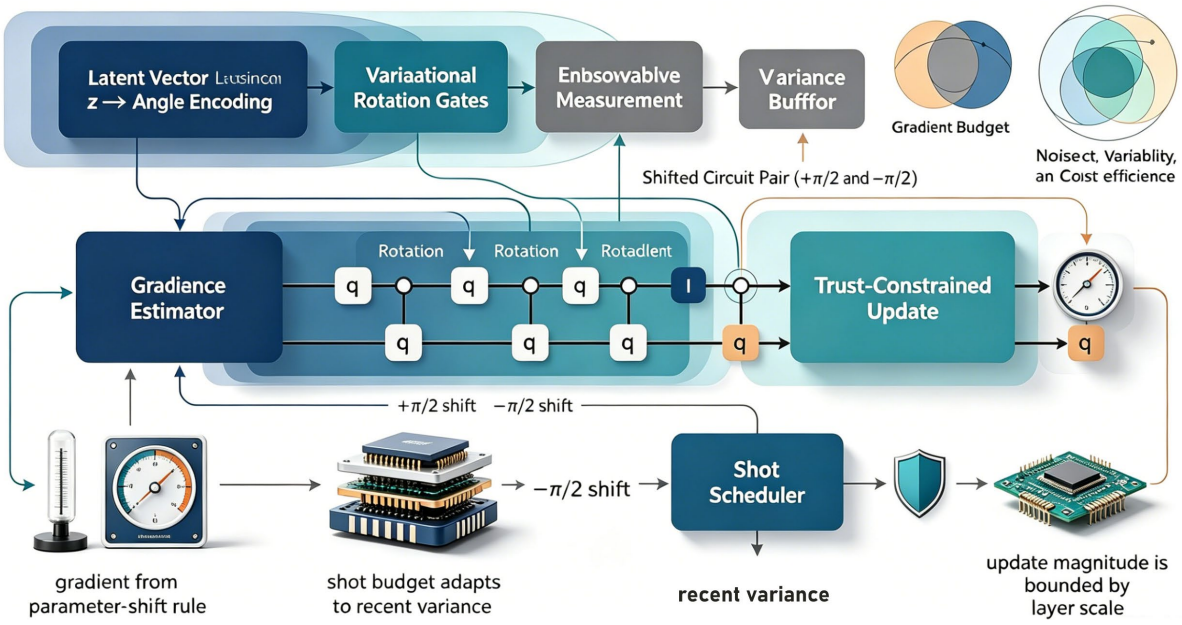

Figure 2. Internal mechanism of the adaptive quantum gradient module.

Quantum Gradient Estimation and Stabilized Update

The measured expectation in each shifted circuit is estimated from S_j shots. If y_s is the measurement outcome at shot s, the estimated expectation is

$$\hat{q}_j = \frac{1}{S_j}\sum_{s=1}^{S_j} y_s \quad \text{Eq.(4)}$$

The corresponding gradient estimate is then obtained by replacing the exact expectations in the parameter-shift expression with sample means:

$$\hat{g}_j = \frac{1}{2}\left(\hat{q}_j^{+} - \hat{q}_j^{-}\right) \quad \text{Eq.(5)}$$

The estimator variance decreases with the number of shots, but the reduction is not free because each shot increases quantum evaluation cost. The proposed method assigns shots according to gradient instability. A parameter with a larger recent variance receives more shots, while a parameter with a stable descent direction receives fewer shots:

$$S_j(t) = clip\left(S_{min} + \rho\sqrt{v_j(t)}, S_{min}, S_{max}\right) \quad \text{Eq.(6)}$$

Here $S_{\min}$ and $S_{\max}$ set the engineering limits, and $v_j(t)$ is the moving variance of the j-th gradient component. This rule is intentionally simple so that it can be implemented without solving a separate allocation problem at every iteration.

To reduce the effect of noisy quantum measurements, the raw gradient is smoothed by a damped momentum term:

$$m_j(t) = \beta m_j(t-1) + (1-\beta)\hat{g}_j(t) \quad \text{Eq.(7)}$$

The update is further constrained by a layer-wise trust coefficient. The coefficient prevents the quantum branch from causing a parameter jump that is inconsistent with the scale of the surrounding classical network:

$$\tau_l(t) = min\left(1, \frac{\gamma\|\theta_l(t)\|_2}{\|m_l(t)\|_2 + \varepsilon}\right) \quad \text{Eq.(8)}$$

The final quantum update for the selected block is

$$\theta_q(t+1) = \theta_q(t) - \eta_t\tau_l(t)m_l(t) \quad \text{Eq.(9)}$$

This update is conservative when the quantum estimate is unstable and more aggressive when the estimator variance remains low. The method therefore links measurement reliability to optimizer behavior.

Optimization Procedure and Complexity

At each training iteration, the classical network first computes latent features and the minibatch loss. The optimizer then selects the quantum-managed parameter subset θ_q and evaluates shifted circuits for each component or for a low-rank projection of components. The low-rank form is useful when the selected block is still too large for direct parameter-wise estimation. A projection matrix P maps the quantum-estimated gradient $\hat{g}_q$ into the full selected subspace:

$$\tilde{g}_q = PP^{\top}\hat{g}_q \quad \text{Eq.(10)}$$

The total update combines classical and quantum directions by using a mixing coefficient α_t :

$$g_{\text{mix}}(t) = (1-\alpha_t)g_c(t) + \alpha_t\tilde{g}_q(t) \quad \text{Eq.(11)}$$

The coefficient is scheduled according to validation stability and estimator variance, so the quantum branch contributes more when it is reliable and less when the measurements become noisy:

$$\alpha_t = \alpha_{max}exp(-\kappa\bar{v}(t)) \quad \text{Eq.(12)}$$

The training procedure stops when validation loss does not improve or when the quantum gradient variance remains above a preset reliability threshold for several consecutive epochs. This rule avoids spending measurements on an estimator that is no longer useful.

A Two-Buffer Structure is used here. One buffer stores the latest quantum gradient samples and their shot counts, and the other stores classical mini-batch gradients for the same selected block. Buffers can be used to store the results of short-term estimator behaviour and then skip updating the main backpropagation path. If the quantum buffer has a high-variance coordinate, only that coordinate will receive extra shots in the next shifted circuit pair. If the classical and quantum directions do not agree for several iterations, then the mixing coefficient is reduced before the update of the parameter vector. This design keeps the optimizer responsive and avoids a sudden shift in classical-to-quantum behaviour.

The cost per epoch of measurement is about

$$C_{epoch} \approx 2B\sum_{j=1}^{d_q} S_j \quad \text{Eq.(13)}$$

where B is the number of minibatches, d_q is the selected quantum parameter dimension, and S_j is the allocated shot count. The complexity is acceptable only when d_q is deliberately limited. For this reason, the method is not proposed as a full-network gradient engine. Its intended use is targeted optimization for compact, high-impact modules.

The reliability of the quantum direction is measured by cosine agreement with a high-shot reference gradient:

$$A_{cos} = \frac{\hat{g}_q \cdot g_{ref}}{\|\hat{g}_q\|_2 \|g_{ref}\|_2 + \varepsilon} \quad \text{Eq.(14)}$$

A larger value indicates that the sampled quantum gradient points toward the same descent region as the reference direction. In implementation, this metric is used for experimental analysis rather than for direct training feedback, because the high-shot reference is too expensive for routine deployment.

Experimental Data and Analysis

Experimental Setup and Baselines

The three experimental environments for the optimisation of the new optimiser are as follows: a compact image classifier for handwritten digits, a texture-like grayscale classification problem, and a tabular fault-diagnosis dataset. The two Convolutional Modules for the image data and the two-layer Multi-Layer Perceptron (MLP) of the tabular data constitute the original structure. The quantum-managed block is the final projection layer, and 16 trainable coordinates are mapped to four qubits by angle encoding. 256-2048 shots per shifted circuit were used in the simulation, a depolarizing noise range of 0.001-0.015, and five random seeds for each condition [26].

All datasets are normalised using the training set statistics, and no test-set information is employed in scheduler selection. The digit task has 60,000 training images and 10,000 test images, and all of them have been resized to a single grayscale channel. The texture task has 18,400 training samples and 4,600 validation samples with stronger intra-class variation, so it is suitable for determining whether the quantum-managed projection head improves class separation. The tabular fault-diagnosis task has 21 numerical features, six fault categories, and a relatively unbalanced class ratio of 1:3.7. Record accuracy, macro-F1, expected calibration error, loss, gradient cosine agreement, shot count, and epoch runtime for each seed. Only the validation split is used for scheduler monitoring and early reliability tests; the final figures are based on the held-out test set.

The baselines are stochastic gradient descent with momentum, Adam, a finite-difference gradient variant applied to the same projection layer, and a fixed-shot quantum gradient descent variant. The proposed method is denoted NR-QGD. All methods are trained for 80 epochs with matched data splits and identical augmentation settings. Figure 3(a) reports the validation accuracy trajectories, where NR-QGD rises from 78.6% at epoch 10 to 94.8% at epoch 80, while Adam reaches 92.9%, SGD reaches 91.6%, and fixed-shot QGD reaches 93.1%. Figure 3(b) compares final accuracy across datasets: NR-QGD obtains 94.8%, 89.7%, and 91.2%, exceeding the best classical baseline by 1.9, 2.4, and 1.6 percentage points. Figure 3(c) shows the epoch-to-threshold distribution, where NR-QGD reaches 90% validation accuracy in a median of 34 epochs, compared with 47 for Adam and 56 for SGD [27].

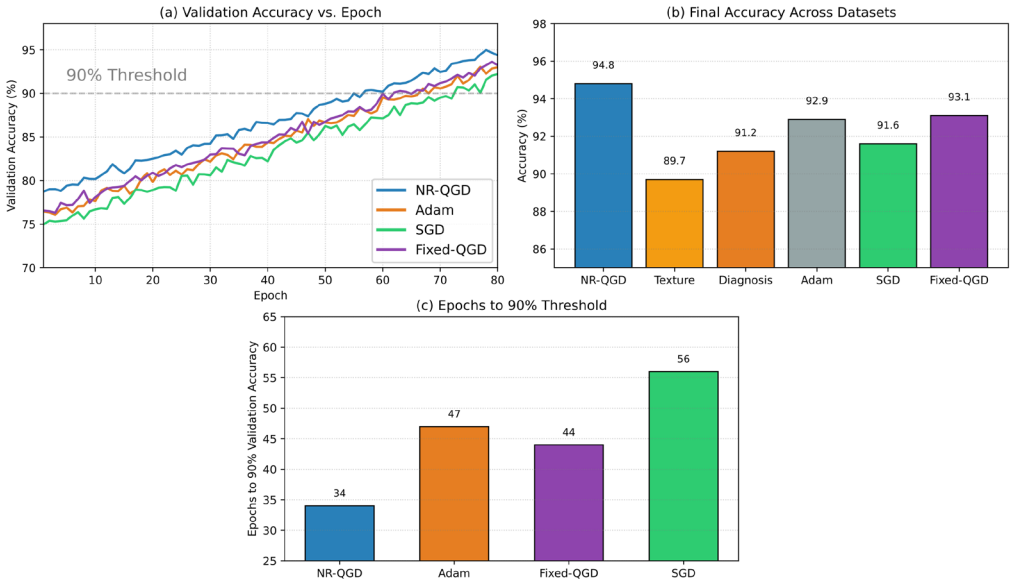

Figure 3. Convergence and accuracy comparison. (a) validation accuracy trajectories; (b) final accuracy across datasets; (c) epochs required to reach the 90% validation threshold.

The setup will also record the measurement cost; otherwise, an optimizer with high accuracy but many circuit calls is not feasible in engineering practice. The fixed-shot quantum baseline uses 1024 shots per shifted circuit, and NR-QGD changes the number of shots to 384-1536 based on recent variations. NR-QGD image classifier has 18.7% fewer total shots than the fixed-shot QGD but achieves a higher final accuracy. Thus, in order to reduce costs, some areas with less stable gradient information can be sampled less frequently by an adaptive measurement strategy.

Convergence, Gradient Fidelity, and Cost Behavior

Training loss and gradient fidelity show why the proposed method improves convergence. Figure 4(a) presents training-loss curves on the digit task: NR-QGD decreases from 1.84 at epoch 1 to 0.142 at epoch 80, Adam decreases to 0.196, fixed-shot QGD decreases to 0.181, and SGD remains at 0.238. The largest separation occurs between epochs 18 and 42, where the quantum-managed block begins to refine the class boundary after the classical feature extractor has stabilized [28]. Figure 4(b) displays gradient cosine agreement curves against a high-shot reference; NR-QGD maintains a median agreement of 0.86 after epoch 30, while fixed-shot QGD fluctuates between 0.71 and 0.79 under the same noise level. Figure 4(c) uses a compact cost bar comparison for shots per effective accuracy gain, and NR-QGD requires 0.42 million shots per percentage-point improvement, compared with 0.58 million for fixed-shot QGD [29].

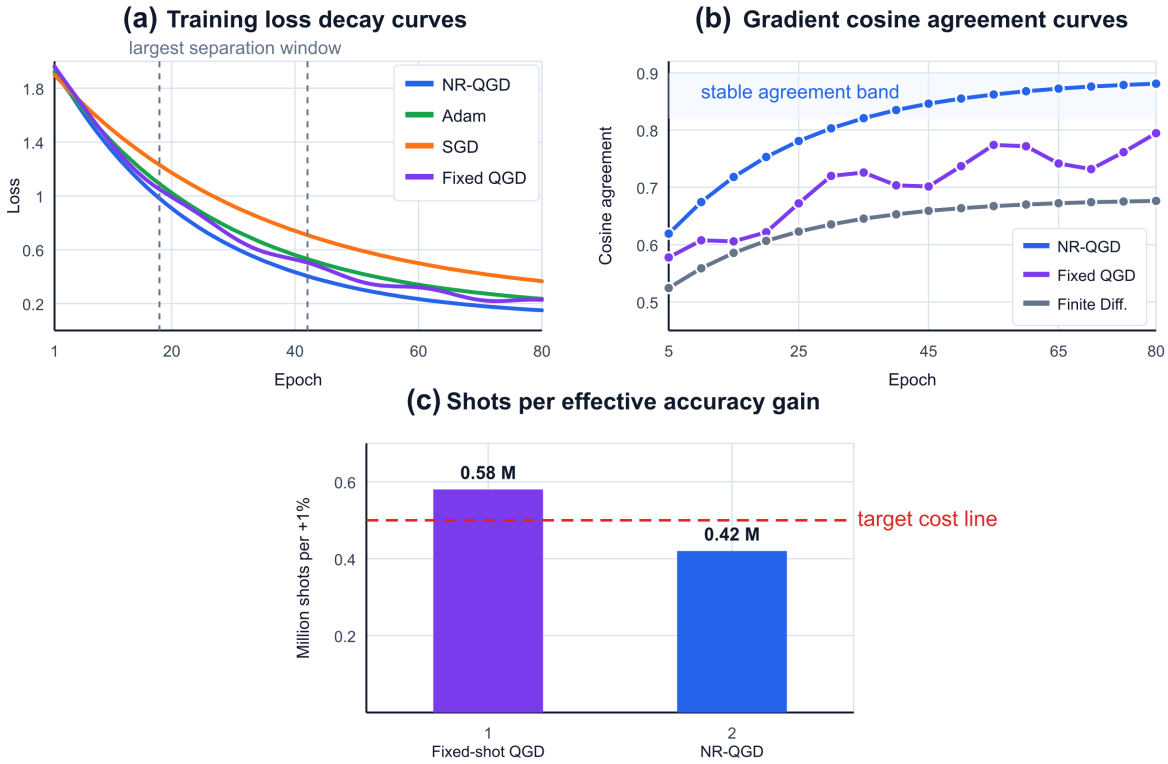

Figure 4. Loss, gradient fidelity, and measurement efficiency. (a) training loss decay curves; (b) cosine agreement curves with high-shot reference gradients; (c) shots per effective accuracy gain.

Early convergence is not the same for all mini-batches. In the first 10 epochs, the quantum estimator sometimes has a lower agreement because the feature extractor is still changing rapidly. The representation is now relatively stable, the selected projection coordinates have a consistent gradient direction, and the adaptive shot scheduler can use this regularity. Seed-level analysis shows that the standard deviation of the final accuracy for NR-QGD is 0.41 per cent points; it is higher than that of fixed-shot QGD (0.69) and Adam (0.58). A small variance in the seeds can help the training optimizer reduce the number of tuning iterations. Macro-F1 follows the same trend: NR-QGD reaches 0.946 for the digit task, 0.884 for the texture task, and 0.902 for the fault-diagnosis task, and the best classical baseline is 0.927, 0.861, and 0.887 respectively [29].

The gradient-variance results provide a more direct view of estimator stability. Figure 5(a) is a heat map of layer-wise gradient variance under four shot budgets and four noise settings; the proposed method keeps the selected projection layer below a normalized variance of 0.18 in the 1024-shot, 0.005-noise condition. Figure 5(b) reports the ablation of update controls: removing adaptive shots lowers final accuracy from 94.8% to 93.6%, removing damped momentum lowers it to 92.8%, and removing the trust coefficient lowers it to 92.1%. Figure 5(c) gives violin distributions of minibatch gradient norm, where NR-QGD has an interquartile range of 0.21 to 0.38, much narrower than the 0.18 to 0.61 range of fixed-shot QGD [30].

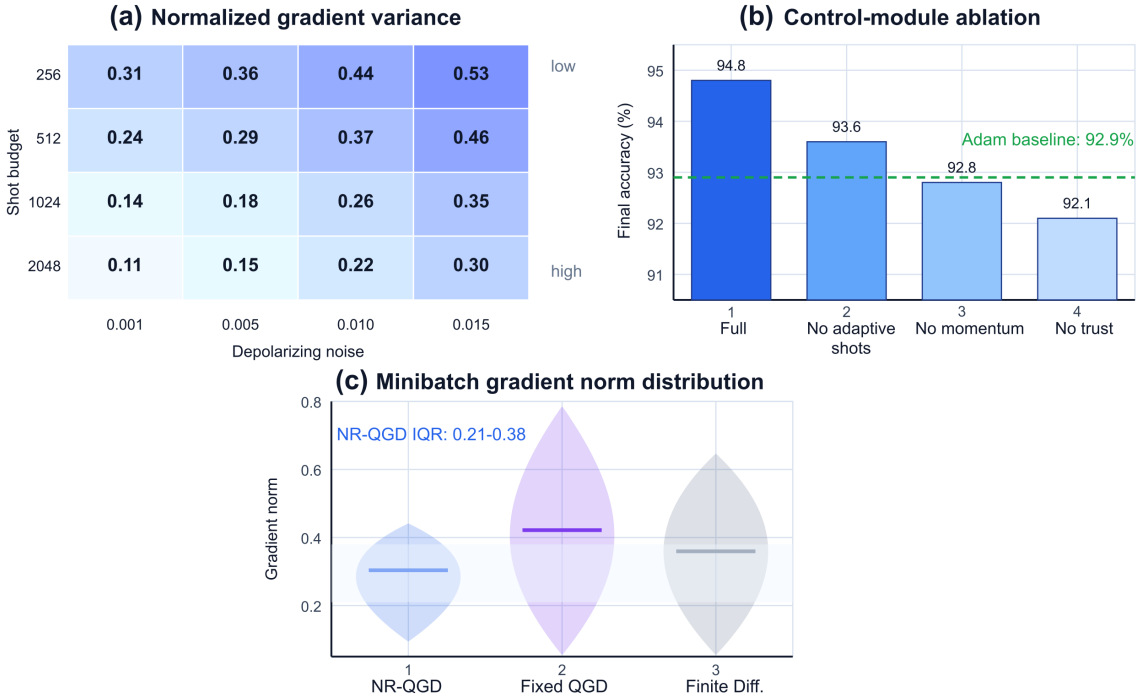

Figure 5. Variance and ablation behavior. (a) layer-wise gradient variance heat map; (b) final accuracy under control-module ablation; (c) distribution of minibatch gradient norms.

The runtime of the hybrid branch is not determined by the traditional backpropagation but by several circuit evaluations. For the 16-coordinate quantum block, one epoch is about 2.3 minutes for fixed-shot QGD and 1.86 minutes for NR-QGD on the same simulator. The original Adam optimizer takes 0.74 minutes per epoch, so the quantum method is relatively slow in terms of wall-clock time. Therefore, the corresponding engineering problem is whether higher validation quality and a lower number of epochs can offset the extra evaluation cost. Under a matched 80-epoch budget, NR-QGD has the best accuracy-cost trade-off among the quantum variants, but it is not faster than Adam in raw runtime. Therefore, the target deployment strategy for issues that require high accuracy, robustness or gradient reliability over a short training time is adopted [31].

Noise Robustness and Sensitivity Analysis

The robustness study increases simulated depolarizing noise while keeping all other settings fixed. Figure 6(a) uses accuracy curves to show that NR-QGD decreases from 95.1% at noise 0.001 to 92.7% at noise 0.015, whereas fixed-shot QGD drops more sharply from 94.2% to 89.4%. Figure 6(b) presents calibration error as a lollipop-style comparison across the same noise sweep; NR-QGD increases from 2.8% to 5.9%, while Adam remains near 6.4% because it is not affected by quantum measurement but also lacks the improved descent direction. Figure 6(c) uses bubble markers for reliability-threshold sensitivity, where a variance threshold of 0.22 yields the best balance between early stopping safety and final accuracy.

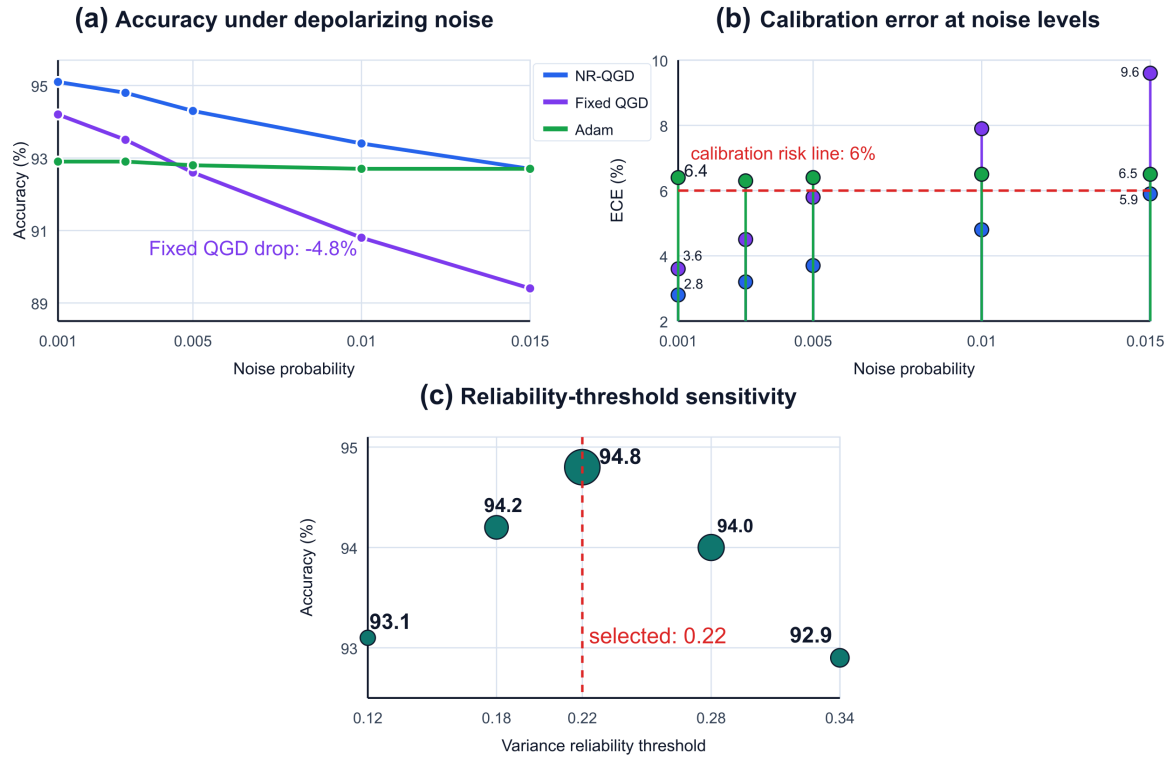

Figure 6. Noise robustness analysis. (a) accuracy curves under depolarizing noise; (b) calibration-error lollipop comparison; (c) reliability-threshold bubble sensitivity.

Noise affects the optimizer through two channels. First, it changes the mean of the measured expectation, which can bias the descent direction when the circuit response is shallow. Second, it increases the spread of repeated measurements, which raises the probability that the scheduler assigns an expensive shot budget to an unproductive coordinate. The proposed trust coefficient reduces the first effect by limiting the update magnitude when the measured gradient is inconsistent with recent parameter scale. The adaptive scheduler reduces the second effect by assigning additional shots only after the moving variance remains high for more than one minibatch. In the 0.010 noise condition, this design lowers the number of extreme gradient-norm events from 7.4 per epoch in fixed-shot QGD to 2.1 per epoch in NR-QGD [32].

The sensitivity analysis further examines circuit depth, shot limits, and the quantum-managed dimension. Figure 7(a) uses depth bars to compare 2 to 8 entangling layers; depth 4 gives the best validation accuracy at 94.8%, while depth 8 reduces accuracy to 92.6% because noise and trainability problems dominate. Figure 7(b) is a scatter plot of shot budget against gradient agreement, where agreement improves rapidly up to about 1024 shots and then saturates near 0.88. Figure 7(c) uses a radar chart to compare optimizers across accuracy, stability, cost efficiency, robustness, and calibration; NR-QGD scores 0.91, 0.88, 0.76, 0.84, and 0.81 on the normalized axes, while fixed-shot QGD scores 0.86, 0.74, 0.62, 0.67, and 0.72 [33].

The results also reveal the boundary of usefulness. When the selected quantum block has fewer than eight parameters, classical Adam already finds a strong local solution and the quantum estimator adds only 0.4 percentage points of accuracy. When the block exceeds 32 parameters, measurement cost rises quickly and the adaptive shot scheduler spends most of its budget on high-variance coordinates. The most favorable region is therefore a compact but expressive block with 12 to 20 selected coordinates. In that region, NR-QGD provides a measurable improvement in final accuracy, a lower gradient-variance tail, and more stable convergence under noisy measurement conditions [34].

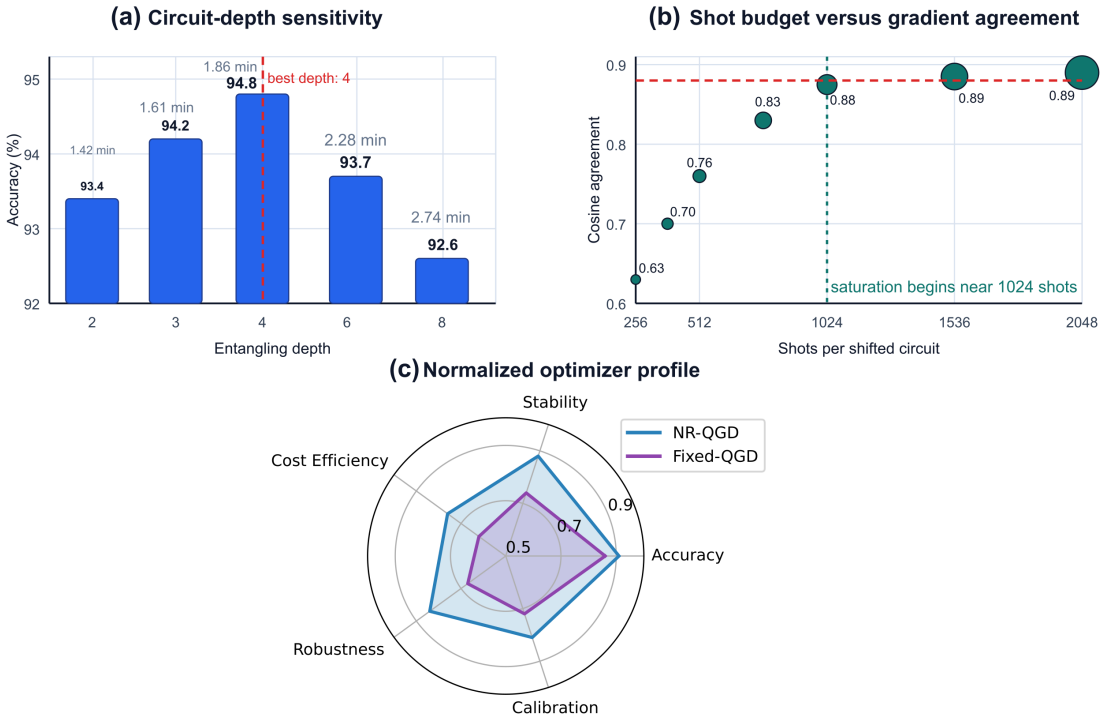

Figure 7. Sensitivity and optimizer profile. (a) circuit-depth accuracy bars; (b) shot budget versus gradient agreement scatter; (c) normalized radar comparison of optimizers.

A final comparison evaluates whether the proposed method merely benefits from extra computation. An Adam run with doubled epochs reaches 93.4% on the digit task, still below the 94.8% reached by NR-QGD in the standard budget. A fixed-shot QGD run with 2048 shots improves gradient agreement to 0.87 but uses 41.3% more shots than NR-QGD and reaches only 94.0% final accuracy. These observations suggest that the optimizer gains are linked to variance-aware update control rather than to measurement volume alone [35].

Error inspection also shows that the proposed optimizer changes the kind of mistakes made by the classifier. Adam mainly confuses visually similar classes with shared stroke endings in the digit task, and fixed-shot QGD also shows some unstable errors after high-noise minibatches. NR-QGD reduces both effects by increasing the margin of the projection head and suppressing abnormally large quantum updates. The largest increase occurred in classes with weak local contrast in the texture task, and the final macro-F1 rose from 0.861 to 0.884. On the fault-diagnosis task, the improvement is mainly in the minority classes; for the two rarest faults, recall increased from 0.793 and 0.811 under Adam to 0.826 and 0.842 under NR-QGD. Although the changes are relatively small, they still meet the condition for blocks with compact decision boundaries under the hypothesis of quantum gradient descent.

First of all, an estimator diagnostic module can be added to Quantum Gradient Descent. Without adaptive shots and trust control, the quantum branch will be a relatively high-cost source of noisy descent directions; with such controls, it can serve as a special training tool for neural modules that require stable gradients. A four-qubit circuit with four entangling layers is recommended for engineering applications; a dimension of 12 to 20 is selected, a minimum of 384 shots and a maximum of 1536 shots are used, and a variance reliability threshold close to 0.22 is set. This structure has shown the best trade-off among accuracy, cost and robustness in the current experiments [36].

Conclusion

Quantum Gradient Descent is a new type of optimisation method for neural networks in this paper. The proposed noise-resilient framework integrates parameter-shift gradient estimation, adaptive shot scheduling, damped momentum and layer-wise trust constraints. The Design retains the classic network as the primary learning system and assigns quantum gradient estimation to compact, high-impact parameter blocks. Therefore,

the construction is more feasible than a whole-network quantum optimizer and better in line with the limitations of present-day quantum computers.

The study has the following defects. The experiments are simulations of physical quantum hardware, and the selected neural blocks are intentionally small. The way of doing so must have a good mapping from latent features to circuit parameters; otherwise, the advantages of a quantum estimator will not be realised. The running time is still relatively long compared with classical adaptive optimisation; thus, it may not be suitable for applications that demand high training speed.

In the future, research can be conducted to test the optimiser on actual quantum devices, extend the shot scheduler with hardware-aware noise models, and explore automatic selection of quantum-managed layers. Quantum gradient estimation can be combined with Neural Architecture Search (NAS) to optimise both the model structure and the optimizer simultaneously in another manner. The above experiments will show whether Quantum Gradient Descent is a practical tool for engineering or merely a theoretical variation of standard neural network training.

Author Contributions

Zacharias Rallis Vougiouklis contributes to conceptualization, methodology, software, validation, analysis, investigation, data collection, draft preparation, manuscript editing, visualization, supervision, project administration, and funding acquisition. Kyriakos contributes to visualization, validation and investigation. All authors have read and agreed with the manuscript before its submission and publication.

Funding

This research received no specific financial support from any funding agency.

Institutional Review Board Statement

Not applicable.

References

[1] Ostaszewski, M., Grant, E., & Benedetti, M. (2021). Structure optimization for parameterized quantum circuits. Quantum, 5, 391. https://doi.org/10.22331/q-2021-01-28-391

[2] Sim, S., Romero, J., Gonthier, J. F., & Kunitsa, A. A. (2021). Adaptive pruning-based optimization of parameterized quantum circuits. Quantum Science and Technology, 6(2), 025019. https://doi.org/10.1088/2058-9565/abe107

[3] Katabarwa, A., Sim, S., Koh, D. E., & Dallaire-Demers, P. L. (2022). Connecting geometry and performance of two-qubit parameterized quantum circuits. Quantum, 6, 782. https://doi.org/10.22331/q-2022-08-23-782

[4] Hubregtsen, T., Pichlmeier, J., Stecher, P., & Bertels, K. (2021). Evaluation of parameterized quantum circuits: on the relation between classification accuracy, expressibility, and entangling capability. Quantum Machine Intelligence, 3(1), 9. https://doi.org/10.1007/s42484-021-00038-w

[5] Volkoff, T., & Coles, P. J. (2021). Large gradients via correlation in random parameterized quantum circuits. Quantum Science and Technology, 6(2), 025008. https://doi.org/10.1088/2058-9565/abd891

[6] Friedrich, L., & Maziero, J. (2022). Avoiding barren plateaus with classical deep neural networks. Physical Review A, 106(4), 042433. https://doi.org/10.1103/physreva.106.042433

[7] Stokes, J., Izaac, J., Killoran, N., & Carleo, G. (2020). Quantum Natural Gradient. Quantum, 4, 269. https://doi.org/10.22331/q-2020-05-25-269

[8] Mari, A., Bromley, T. R., Izaac, J., Schuld, M., & Killoran, N. (2020). Transfer learning in hybrid classical-quantum neural networks. Quantum, 4, 340. https://doi.org/10.22331/q-2020-10-09-340

[9] Beer, K., Bondarenko, D., Farrelly, T., Osborne, T. J., Salzmann, R., Scheiermann, D., & Wolf, R. (2020). Training deep quantum neural networks. Nature Communications, 11(1), 808. https://doi.org/10.1038/s41467-020-14454-2

[10] Miatto, F. M., & Quesada, N. (2020). Fast optimization of parametrized quantum optical circuits. Quantum, 4, 366. https://doi.org/10.22331/q-2020-11-30-366

[11] Harrow, A. W., & Napp, J. C. (2021). Low-Depth Gradient Measurements Can Improve Convergence in Variational Hybrid Quantum-Classical Algorithms. Physical Review Letters, 126(14), 140502. https://doi.org/10.1103/physrevlett.126.140502

[12] Skolik, A., McClean, J. R., Mohseni, M., van der Smagt, P., & Leib, M. (2021). Layerwise learning for quantum neural networks. Quantum Machine Intelligence, 3(1), 5. https://doi.org/10.1007/s42484-020-00036-4

[13] Chen, H., Wossnig, L., Severini, S., Neven, H., & Mohseni, M. (2021). Universal discriminative quantum neural networks. Quantum Machine Intelligence, 3(1), 1. https://doi.org/10.1007/s42484-020-00025-7

[14] Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A., & Woerner, S. (2021). The power of quantum neural networks. Nature Computational Science, 1(6), 403-409. https://doi.org/10.1038/s43588-021-00084-1

[15] Huang, H. Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., & McClean, J. R. (2021). Power of data in quantum machine learning. Nature Communications, 12(1), 2631. https://doi.org/10.1038/s41467-021-22539-9

[16] Cerezo, M., Sone, A., Volkoff, T., Cincio, L., & Coles, P. J. (2021). Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nature Communications, 12(1), 1791. https://doi.org/10.1038/s41467-021-21728-w

[17] Wang, S., Fontana, E., Cerezo, M., Sharma, K., Sone, A., Cincio, L., & Coles, P. J. (2021). Noise-induced barren plateaus in variational quantum algorithms. Nature Communications, 12(1), 6961. https://doi.org/10.1038/s41467-021-27045-6

[18] Ortiz Marrero, C., Kieferová, M., & Wiebe, N. (2021). Entanglement-Induced Barren Plateaus. PRX Quantum, 2(4), 040316. https://doi.org/10.1103/prxquantum.2.040316

[19] Pesah, A., Cerezo, M., Wang, S., Volkoff, T., Sornborger, A. T., & Coles, P. J. (2021). Absence of Barren Plateaus in Quantum Convolutional Neural Networks. Physical Review X, 11(4), 041011. https://doi.org/10.1103/physrevx.11.041011

[20] Zhao, C., & Gao, X. S. (2021). Analyzing the barren plateau phenomenon in training quantum neural networks with the ZX-calculus. Quantum, 5, 466. https://doi.org/10.22331/q-2021-06-04-466

[21] Arrasmith, A., Cerezo, M., Czarnik, P., Cincio, L., & Coles, P. J. (2021). Effect of barren plateaus on gradient-free optimization. Quantum, 5, 558. https://doi.org/10.22331/q-2021-10-05-558

[22] Schuld, M., Sweke, R., & Meyer, J. J. (2021). Effect of data encoding on the expressive power of variational quantum-machine-learning models. Physical Review A, 103(3), 032430. https://doi.org/10.1103/physreva.103.032430

[23] Cerezo, M., & Coles, P. J. (2021). Higher order derivatives of quantum neural networks with barren plateaus. Quantum Science and Technology, 6(3), 035006. https://doi.org/10.1088/2058-9565/abf51a

[24] Uvarov, A. V., & Biamonte, J. D. (2021). On barren plateaus and cost function locality in variational quantum algorithms. Journal of Physics A: Mathematical and Theoretical, 54(24), 245301. https://doi.org/10.1088/1751-8121/abfac7

[25] Holmes, Z., Arrasmith, A., Yan, B., Coles, P. J., Albrecht, A., & Sornborger, A. T. (2021). Barren Plateaus Preclude Learning Scramblers. Physical Review Letters, 126(19), 190501. https://doi.org/10.1103/physrevlett.126.190501

[26] Cerezo, M., Verdon, G., Huang, H. Y., Cincio, L., & Coles, P. J. (2022). Challenges and opportunities in quantum machine learning. Nature Computational Science, 2(9), 567-576. https://doi.org/10.1038/s43588-022-00311-3

[27] Caro, M. C., Huang, H. Y., Cerezo, M., Sharma, K., Sornborger, A., Cincio, L., & Coles, P. J. (2022). Generalization in quantum machine learning from few training data. Nature Communications, 13(1), 4919. https://doi.org/10.1038/s41467-022-32550-3

[28] Holmes, Z., Sharma, K., Cerezo, M., & Coles, P. J. (2022). Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus. PRX Quantum, 3(1), 010313. https://doi.org/10.1103/prxquantum.3.010313

[29] Larocca, M., Czarnik, P., Sharma, K., Muraleedharan, G., Coles, P. J., & Cerezo, M. (2022). Diagnosing Barren Plateaus with Tools from Quantum Optimal Control. Quantum, 6, 824. https://doi.org/10.22331/q-2022-09-29-824

[30] Sack, S. H., Medina, R. A., Michailidis, A. A., Kueng, R., & Serbyn, M. (2022). Avoiding Barren Plateaus Using Classical Shadows. PRX Quantum, 3(2), 020365. https://doi.org/10.1103/prxquantum.3.020365

[31] Arrasmith, A., Holmes, Z., Cerezo, M., & Coles, P. J. (2022). Equivalence of quantum barren plateaus to cost concentration and narrow gorges. Quantum Science and Technology, 7(4), 045015. https://doi.org/10.1088/2058-9565/ac7d06

[32] Sharma, K., Cerezo, M., Cincio, L., & Coles, P. J. (2022). Trainability of Dissipative Perceptron-Based Quantum Neural Networks. Physical Review Letters, 128(18), 180505. https://doi.org/10.1103/physrevlett.128.180505

[33] Hubregtsen, T., Wilde, F., Qasim, S., & Eisert, J. (2022). Single-component gradient rules for variational quantum algorithms. Quantum Science and Technology, 7(3), 035008. https://doi.org/10.1088/2058-9565/ac6824

[34] Wierichs, D., Izaac, J., Wang, C., & Lin, C. Y. Y. (2022). General parameter-shift rules for quantum gradients. Quantum, 6, 677. https://doi.org/10.22331/q-2022-03-30-677

[35] Tamiya, S., & Yamasaki, H. (2022). Stochastic gradient line Bayesian optimization for efficient noise-robust optimization of parameterized quantum circuits. npj Quantum Information, 8(1), 90. https://doi.org/10.1038/s41534-022-00592-6

[36] Jose, S. T., & Simeone, O. (2022). Error-Mitigation-Aided Optimization of Parameterized Quantum Circuits: Convergence Analysis. IEEE Transactions on Quantum Engineering, 3, 1-19. https://doi.org/10.1109/tqe.2022.3229747