A TRAINABLE CLOSED-FORM QUANTUM NEURON: PARAMETER EFFICIENCY AND TRAINABILITY AT SCALE

Anonymous authors
Paper under double-blind review

ABSTRACT

Variational quantum neural networks (VQNNs) offer attractive theoretical properties: periodic response functions arising from quantum interference provide non-linear expressive power, and resource requirements can scale linearly with input dimension. However, training is costly and difficult to deploy, and most architectures lack an atomic learning primitive whose input–output map admits a closed form for rigorous analysis. We propose the Variational Quantum Neuron (VQN), a single-neuron-level closed-form solution that we prove is mathematically equivalent to its corresponding quantum circuit, and verify this equivalence independently via circuit simulation to within floating-point precision. This closed form enables training entirely without circuit simulation, at negligible computational cost. We show that a single VQN, using only $m + 1$ parameters, achieves 100% accuracy on the m-bit parity task for m up to 20, while a parameter-matched perceptron remains at chance level; matching VQN's performance with classical ReLU networks given substantially larger capacity requires over 135 times more parameters at $m = 16$, with this gap widening sharply as task scale increases. Our results demonstrate that a rigorously derived, quantum-circuit-equivalent closed form can serve as a parameter-efficient, readily trainable learning primitive at practical scales.

1 INTRODUCTION

Variational Quantum Neural Networks (VQNNs) have emerged as a leading paradigm in quantum machine learning (Farhi & Neven, 2018; Mitarai et al., 2018; Benedetti et al., 2019; Cerezo et al., 2021a). In principle, VQNNs offer attractive theoretical properties: quantum superposition and interference naturally give rise to periodic response functions (Schuld et al., 2021), endowing the model with non-linear expressive power (Mitarai et al., 2018; Pérez-Salinas et al., 2020); and the resource requirements can be designed to scale linearly with the input dimension.

In practice, however, training such networks remains costly and difficult to deploy (Mitarai et al., 2018; Schuld et al., 2019; Wierichs et al., 2022; McClean et al., 2018; Cerezo et al., 2021b; Holmes et al., 2022). Most existing VQNN architectures rely on full quantum circuit simulation, whose computational cost grows steeply with problem size. Moreover, most VQNN designs lack an atomic quantum learning unit that is analytically tractable in the way a classical neuron is (Schuld et al., 2014): one cannot, in general, write down a closed-form input–output mapping (Cao et al., 2017; Moreira et al., 2023; Tacchino et al., 2019), which makes rigorous analysis of training behavior difficult.

To address this gap, we propose the Variational Quantum Neuron (VQN): a single-neuron-level closed-form solution that we prove is mathematically equivalent to the computation performed by its corresponding quantum circuit. Specifically, we show that the neuron's scalar output admits the exact closed form

$$r(x, \theta) = -\sum_{s \in \{0,1\}^m} p_s(x) \cos(\theta_r(s)), \tag{1}$$

where $x \in \mathbb{R}^m$ is the input, $\theta \in \mathbb{R}^{m+1}$ are the trainable parameters, $p_s(x)$ are data-dependent path probabilities, and $\theta_r(s) = \sum_j s_j \theta_j + \theta_{m+1}$ are the branch-conditioned phases (Theorem 1). This

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

closed form allows the neuron to be trained entirely without circuit simulation, using standard deep learning frameworks (Paszke et al., 2019). We verify this equivalence independently via PennyLane (Bergholm et al., 2018) circuit simulation, confirming that the closed-form output matches the true quantum circuit output to within floating-point precision.

Beyond its analytical tractability, we show that this closed form possesses expressive power exceeding that of monotonic activation functions such as ReLU—a natural consequence of its periodic structure that overcomes the classical limitation of linear perceptrons on non-linearly separable functions (Minsky & Papert, 2017). On the m-bit parity task (Farhi & Neven, 2018; Cao et al., 2017) for $m = 2$ to 20, a single VQN with only $m + 1$ parameters achieves 100% accuracy, while a parameter-matched perceptron remains at chance level throughout. Furthermore, when compared against parameter-*unmatched* classical ReLU networks—allowing the classical model arbitrarily more capacity—we find that matching VQN's performance requires substantially more parameters: approximately $16\times$, $19\times$, and $71\times$ more parameters at $m = 2$, 4, and 8 respectively; at $m = 16$, even a classical network with over $135\times$ VQN's parameter count fails to match its performance. This indicates that VQN's parameter efficiency advantage widens as task scale increases.

Our contributions are as follows:

- We prove, in closed mathematical form, and independently verify via circuit simulation, the exact input–output equivalence between a single VQN and its corresponding quantum circuit.
- We show that this closed form enables training entirely without quantum circuit simulation, at a low computational cost (single training runs complete in approximately 0.1–0.3 seconds), making it directly deployable with standard deep learning tools.
- We systematically characterize the parameter efficiency of this closed-form neuron relative to classical neurons, including classical networks given substantially larger parameter budgets.

2 Related Work

Our proposal touches two bodies of work: (i) quantum neurons and quantum perceptrons, which define the design space our closed-form construction is compared against; and (ii) expressivity analyses of periodic response functions, both quantum and classical, which bear directly on the question of where the advantage we observe originates.

2.1 Quantum neurons and quantum perceptrons

The idea of a quantum analogue of the classical neuron predates the current variational-circuit era. Early quantum-perceptron proposals in which quantum superposition replaces the classical dot product date back to Kak (1995) and Altaisky (2001); Schuld et al. (2014) subsequently reviewed the design space and identified a persistent tension: introducing a genuine nonlinearity into a unitary circuit, which is necessary for any neuron-like activation, requires either measurement collapse, ancillary post-selection, or a repeat-until-success (RUS) construction (Cao et al., 2017). Tacchino et al. (2019) proposed a memory-efficient binary quantum perceptron encoding classical inputs and weights as sign patterns of m-qubit computational-basis states, achieving a proof of concept on IBM hardware but restricted to ± 1-valued weights; a superconducting repeat-until-success realization of the RUS neuron was later demonstrated by Moreira et al. (2023). A distinct, unitary-dissipative line by Beer et al. (2020) trained layer-wise quantum perceptrons at the cost of fresh ancilla qubits per layer and without a closed-form response. More broadly, circuit-centric variational classifiers built from hardware-efficient or data re-uploading layers (Schuld et al., 2020; Havlíček et al., 2019; Pérez-Salinas et al., 2020) have become the dominant template for supervised VQNNs, but their expressivity and trainability are typically established empirically rather than through a closed-form input–output map.

The proposed VQN differs from these constructions on three axes. First, the nonlinearity is *deterministic*, arising from the Hadamard-flanked interferometer rather than from post-selection or repeat-until-success (Cao et al., 2017; Moreira et al., 2023), so the readout expectation is a fixed

function of (x, θ) with no failed attempts or discarded runs. Second, the input encoding is real-valued angle encoding (Schuld & Killoran, 2019), not sign encoding, so the model handles continuous features directly, requiring only standard min–max normalization (equation 3). Third, the input–output relation is available in closed form (equation 1).

2.2 Expressivity: quantum and classical periodic constructions

A PQC that encodes classical data through Pauli rotations can, under mild assumptions, be written as a real trigonometric polynomial in the inputs; the set of accessible frequencies, known as the *Fourier spectrum*, determines the class of functions the circuit can represent (Schuld et al., 2021). From this viewpoint, since each feature is encoded by a single Pauli rotation, the VQN's accessible frequency vectors with respect to the inputs are $\{-1, 0, 1\}^m$ (in units of π), i.e., 3^m frequencies obtained without any repeated data uploading (Appendix A.9.2). Schreiber et al. (2023) further showed that broad classes of such trigonometric models admit efficient *classical surrogates*, sharpening the notion of what would count as a genuine quantum advantage in expressivity.

This concern is not merely theoretical: periodic activation functions are known, independently of any quantum motivation, to endow a single classical neuron with expressive power beyond that of monotonic units such as ReLU or the classical perceptron. Mallick et al. (2020) prove that a three-neuron periodic perceptron can realize an arbitrary Boolean function and demonstrate this concretely on the XOR and parity tasks, outperforming standard multilayer perceptrons of matched size. More recently, Huang & Li (2025) prove that a width-2 sinusoidal MLP exactly realizes modular addition—a generalization of k-parity—for any input length, whereas a ReLU network requires width growing linearly with the input length and cannot simultaneously fit two different lengths; they further show that substituting sinusoidal activations into a Transformer's feed-forward layers improves sample efficiency on the same task. Our closed-form neuron therefore does not introduce periodicity as a novel source of expressive power; rather, its contribution is to derive a specific periodic form as the exact, provable output of a physically motivated quantum circuit. This circuit correspondence, which lies outside the scope of the classical constructions of Mallick et al. and Huang & Li, yields a quantum resource accounting ($m + 1$ qubits and m two-qubit gates; Remark A.1) and guarantees that gradients computed classically through the closed form coincide exactly with those obtained on the circuit via the parameter-shift rule (Appendix A.10), so that parameters trained classically transfer directly to quantum hardware.

This distinction also bears on trainability. McClean et al. (2018) showed that for random deep parameterized circuits acting on n qubits, the variance of any observable's gradient decays exponentially in n, a phenomenon known as the *barren plateau*; Cerezo et al. (2021b) refined this to a cost-function dependence, showing that shallow circuits with local costs need not exhibit exponential decay. Because our neuron's closed form factorizes over features, its gradient variance at random initialization can be computed exactly (Appendix A.11). For binary inputs, the setting of all our experiments, the variance does not decay with m at all, so the degradation we observe beyond $m \approx 20$ (Section 4.3) is not caused by vanishing gradients at initialization. For continuous inputs, by contrast, the variance decays exponentially in the number of non-binary features at a fixed initialization scale, although shrinking the initialization scale with m may counteract this.

3 The Variational Quantum Neuron

A classical neuron takes m input values $\mathbf{x} \equiv (x_1, \ldots, x_m) \in \mathbb{R}^m$ and weight values $\mathbf{w} \equiv (w_1, \ldots, w_m) \in \mathbb{R}^m$, together with a bias w_{m+1} associated with a fixed input $x_{m+1} (= 1.0)$. The neuron computes the dot product of the input and weight vectors plus the bias, and passes this net sum through a function f to produce the activation $a \equiv f\big(\sum_{j=1}^{m+1} x_j w_j\big)$. The function f is normally *monotonic* (e.g., sigmoid, tanh, or ReLU); it is this monotonicity that renders a single such neuron incapable of representing non-linearly separable functions such as parity (Minsky & Papert, 2017).

Rather than heuristically claiming a generic quantum computational advantage, we show that a specific, physically motivated quantum circuit replaces the classical monotonic activation with a deterministic, periodic one, and that the resulting input–output map admits an exact closed form. We define the Variational Quantum Neuron (VQN) operationally as a parameterized quantum circuit

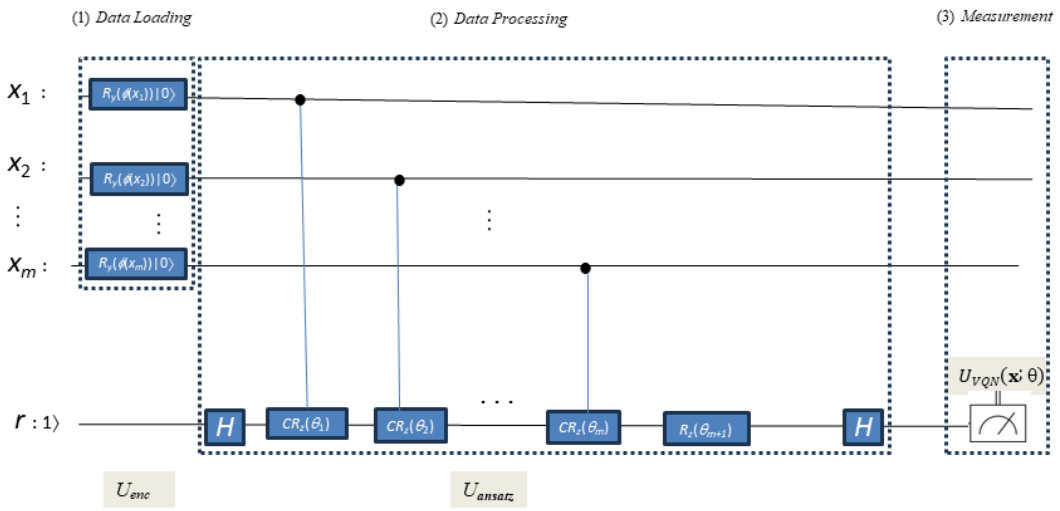


Figure 1: The quantum circuit of the proposed VQN.

Table 1: Notation used in this section.

Symbol	Meaning
$\mathbf{x} \equiv (x_1, \ldots, x_m)^\top, x_j \in \mathbb{R}$	Input feature vector.
m	Number of input features.
$\theta_j, j = 1, \ldots, m$	Weight associated with input feature x_j.
θ_{m+1}	Bias parameter.
$\boldsymbol{\theta} \equiv \{\theta_1, \ldots, \theta_{m+1}\} \in \mathbb{R}^{m+1}$	Full trainable parameter vector.
r	Readout qubit (wire $m+1$).
$\mathbf{s} \equiv (s_1, \ldots, s_m) \in \{0,1\}^m$	Computational-basis bitstring (branch) of the input register.
$q_{\mathbf{s}}(\mathbf{x}), p_{\mathbf{s}}(\mathbf{x}) \equiv q_{\mathbf{s}}(\mathbf{x})^2$	Path amplitude and path probability of branch $\mathbf{s}$.
$\theta_r(\mathbf{s})$	Phase accumulated on the readout qubit in branch $\mathbf{s}$.
$z(\mathbf{x}, \boldsymbol{\theta}) \equiv \langle Z_r \rangle_{\mathbf{x},\boldsymbol{\theta}}$	Scalar readout of the VQN.
β	Sharpness hyperparameter of the soft decision rule.
τ	Decision threshold.
$\sigma(\cdot)$	Logistic sigmoid, $\sigma(u) = 1/(1+e^{-u})$.
K	Number of training samples.

(PQC) that maps a classical feature vector to a scalar measurement outcome through data encoding, a trainable unitary of fixed structure, and a single-qubit measurement.

Fig. 1 shows the quantum circuit of the proposed VQN. The construction consists of three stages: (1) data loading via a feature map $U_{\text{enc}}(\mathbf{x})$; (2) a fixed-structure ansatz $U_{\text{ansatz}}(\boldsymbol{\theta})$ parameterized by trainable weights $\boldsymbol{\theta} \equiv \{\theta_1, \ldots, \theta_m, \theta_{m+1}\} \in \mathbb{R}^{m+1}$, where θ_j is the weight of input feature x_j and θ_{m+1} is the bias; and (3) a measurement-based readout on a dedicated readout qubit r. Table 1 lists the notation used throughout this section.

Analogous to a classical neuron, the proposed VQN maps an input vector to a scalar readout, but the mechanism differs fundamentally: instead of the classical linear weighted sum $\sum_{j=1}^{m} w_j x_j + w_{m+1}$ passed through a monotonic activation, the VQN implements a branch-conditioned phase aggregation followed by a Hadamard-induced interference readout. This yields a shallow, parameter-efficient nonlinear mapping governed entirely by quantum interference, with no separate classical activation function. We summarize the derivation below; a complete step-by-step proof is given in Appendix A.

3.1 CIRCUIT CONSTRUCTION

The VQN circuit uses $m+1$ qubits, all initialized to $|0\rangle$: wires $1,\dots,m$ encode the input features, and wire $r=m+1$ is the readout. Thus $|\psi_0\rangle = |0\rangle^{\otimes(m+1)}$.

The data-loading unitary is

$$U_{\mathrm{enc}}(\mathbf{x}) = \bigotimes_{j=1}^{m} R_y\big(\pi x_j\big) \otimes X_r, \tag{2}$$

where $R_y(\pi x_j)|0\rangle = \cos(\pi x_j/2)|0\rangle + \sin(\pi x_j/2)|1\rangle$ angle-encodes each feature $x_j \in [0,1]$, and Pauli-X prepares the readout in $|1\rangle$. Continuous features are first mapped to $[0,1]$ by per-feature min–max normalization,

$$x_j \leftarrow \frac{x_j - x_j^{\min}}{x_j^{\max} - x_j^{\min}}, \tag{3}$$

where $x_j^{\min}$ and $x_j^{\max}$ are the minimum and maximum values of feature j.

For each bitstring $\mathbf{s} \equiv (s_1,\dots,s_m) \in \{0,1\}^m$, define the path amplitude

$$q_{\mathbf{s}}(\mathbf{x}) \equiv \prod_{j=1}^{m} \Big[\cos(\pi x_j/2)^{1-s_j} \cdot \sin(\pi x_j/2)^{s_j}\Big], \tag{4}$$

so that

$$|\psi_{\mathrm{enc}}\rangle \equiv U_{\mathrm{enc}}(\mathbf{x})|\psi_0\rangle = \sum_{\mathbf{s}\in\{0,1\}^m} q_{\mathbf{s}}(\mathbf{x})\, |s_1\cdots s_m\rangle_{1,\dots,m} \otimes |1\rangle_r. \tag{5}$$

The ansatz aggregates feature-conditioned phases onto the readout qubit through CR_z and R_z rotations, flanked by two Hadamard gates. The full VQN unitary is

$$\begin{aligned} U_{\mathrm{VQN}}(\mathbf{x},\boldsymbol{\theta}) &\equiv U_{\mathrm{ansatz}}(\boldsymbol{\theta})\, U_{\mathrm{enc}}(\mathbf{x}), \\ U_{\mathrm{ansatz}}(\boldsymbol{\theta}) &\equiv H_r\, R_z(\theta_{m+1})\Big(\prod_{j=1}^{m} \mathrm{CR}_z^{(j,r)}(\theta_j)\Big) H_r. \end{aligned} \tag{6}$$

All CR_z gates in the product share the same target r and mutually commute (Remark A.1), so the product is well defined and the gates may be applied in any order. The phase accumulated on the readout qubit in branch $\mathbf{s}$ is

$$\theta_r(\mathbf{s}) = \sum_{j=1}^{m} s_j\, \theta_j + \theta_{m+1}. \tag{7}$$

The bias θ_{m+1} is realized as an R_z rotation on the readout wire rather than a separate bias qubit. The final Hadamard converts the accumulated relative phase into a measurable population difference via constructive and destructive interference.

Upon measurement, the scalar readout of the VQN is

$$z(\mathbf{x},\boldsymbol{\theta}) \equiv \langle Z_r\rangle_{\mathbf{x},\boldsymbol{\theta}} = \langle\psi_0|U_{\mathrm{VQN}}^{\dagger}(\mathbf{x},\boldsymbol{\theta})\, Z_r\, U_{\mathrm{VQN}}(\mathbf{x},\boldsymbol{\theta})|\psi_0\rangle = P_r(0) - P_r(1) \in [-1,1], \tag{8}$$

where $P_r(b)$ is the probability of measuring the readout qubit in $|b\rangle$. Section 3.4 describes how $z(\mathbf{x},\boldsymbol{\theta})$ is converted into a class prediction.

3.2 CLOSED-FORM SOLUTION

Theorem 1 (Closed-form VQN mapping)**.** *Let $U_{\mathrm{VQN}}(\mathbf{x},\boldsymbol{\theta})$ be defined by equation 6, and let $p_{\mathbf{s}}(\mathbf{x}) \equiv q_{\mathbf{s}}(\mathbf{x})^2$. Then the scalar readout $z(\mathbf{x},\boldsymbol{\theta})$ of equation 8 admits the exact closed form*

$$z(\mathbf{x},\boldsymbol{\theta}) = -\sum_{\mathbf{s}\in\{0,1\}^m} p_{\mathbf{s}}(\mathbf{x}) \cdot \cos\big(\theta_r(\mathbf{s})\big). \tag{9}$$

The nonlinearity is thus entirely induced by interference at the readout, rather than by an ad hoc classical activation function. The proof (Appendix A) proceeds by propagating the branch-factorized state through each stage of U_{VQN} and evaluating $\langle Z_r \rangle$ branch-by-branch.

We verify Theorem 1 independently by comparing equation 9, evaluated directly in PyTorch, against a full state-vector simulation of the circuit in PennyLane over 30 random draws of $(\mathbf{x}, \boldsymbol{\theta})$ for each $m = 2, \ldots, 6$; the maximum absolute deviation is at most 7.4×10^{-16}, at the level of double-precision round-off (Section 4.2). This confirms that the closed form is not an approximation but an exact reformulation of the circuit's output. Consequently, all experiments in this paper are conducted by training the closed form of equation 9 directly in PyTorch: no quantum circuit simulation is used beyond this one-time equivalence check, so the cost of training is not tied to the cost of simulating the circuit.

Although equation 9 is written as a sum over 2^m branches, it factorizes over features. With $a_j \equiv \sin^2(\pi x_j/2)$, both $p_{\mathbf{s}}(\mathbf{x})$ and $e^{i\theta_r(\mathbf{s})}$ are products of per-feature terms, so that

$$z(\mathbf{x}, \boldsymbol{\theta}) = -\operatorname{Re}\left[e^{i\theta_{m+1}} \prod_{j=1}^{m} \left(1 - a_j + a_j e^{i\theta_j}\right) \right]. \tag{10}$$

The closed form can therefore be evaluated classically in $O(m)$ operations for any input, binary or continuous (Appendix A.9.3).

3.3 Boundary reduction for binary inputs

When $x_j \in \{0, 1\}$ for all j, we have $\cos(\pi \cdot 0/2) = 1$ and $\sin(\pi \cdot 1/2) = 1$, so $q_{\mathbf{s}}(\mathbf{x}) = \delta_{\mathbf{s},\mathbf{x}}$ and equation 9 collapses to a single term:

$$z(\mathbf{x}, \boldsymbol{\theta})\big|_{\text{binary}} = -\cos\Big(\sum_{j=1}^{m} x_j \theta_j + \theta_{m+1} \Big), \quad x_j \in \{0, 1\}. \tag{11}$$

This is the form used throughout our experiments on the m-bit parity task (Sections 4.3 and 4.4): a single cosine of a linear combination of binary inputs, evaluated and trained without any circuit simulation.

3.4 Training procedure

For binary classification, the readout $z(\mathbf{x}, \boldsymbol{\theta}) \in [-1, 1]$ of equation 9 is converted to a soft decision via

$$\hat{y}(\mathbf{x}, \boldsymbol{\theta}) = \sigma\big(\beta\,(|z(\mathbf{x}, \boldsymbol{\theta})| - \tau)\big), \tag{12}$$

where τ is the decision threshold (set to 0.5 throughout) and $\beta > 0$ is a sharpness hyperparameter controlling the steepness of the decision boundary. This rule is invariant under $z \to -z$: it assigns class 1 when the readout is strongly polarized ($|z|$ close to 1) and class 0 when the readout is close to zero. Because $|z|$ is π-periodic in the aggregated phase, a single neuron can represent parity exactly. In the binary case of equation 11, setting $\theta_j = \pi/2$ for all j and $\theta_{m+1} = 0$ gives $|z| = 1$ on inputs with even Hamming weight and $z = 0$ on inputs with odd Hamming weight, while setting $\theta_{m+1} = \pi/2$ exchanges the two cases; parity is therefore exactly realizable under either label convention.

Parameters are trained by minimizing the mean squared error $\frac{1}{K}\sum_{k=1}^{K}(\hat{y}^{(k)} - y^{(k)})^2$ via Adam, with gradients obtained by standard automatic differentiation of equation 12 through the closed form. No parameter-shift evaluation or circuit re-execution is required, since $z(\mathbf{x}, \boldsymbol{\theta})$ is already an explicit, differentiable function of $\boldsymbol{\theta}$.

Because we found training to be sensitive to both learning rate and batch size, we select both jointly via a grid search on a held-out set of random seeds before final evaluation; the search protocol is described in Section 4.1.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We use the m-bit parity task, which has a finite input space of 2^m patterns, as our dataset and benchmark. For $m \leq 20$, we train and evaluate on the complete set of patterns, so that the reported test accuracy coincides with training accuracy and reflects performance over the entire input space; in this regime, the experiments measure a model's ability to represent and fit the parity function rather than its ability to generalize to unseen inputs. For $m > 20$, the input space becomes too large to enumerate within our training budget, so we sample a fixed training set and a held-out test set independently for each random seed and report test accuracy on the held-out set.

All experiments were conducted on a single NVIDIA A100 GPU (80 GB). Following Section 3.2, every model in this section—the proposed VQN, the classical perceptron, and all classical multi-layer baselines—is trained entirely in PyTorch using standard automatic differentiation; no quantum circuit simulator is used for training. The sole exception is the equivalence check in Section 4.2, which uses PennyLane once for verification.

Because we found training to be sensitive to both learning rate and batch size, we select both jointly for every model and every task configuration via a two-stage protocol: a coarse grid search over a candidate set of learning rates and batch sizes (evaluated on a small held-out set of random seeds), followed by full evaluation of the best-performing configuration on 40 independent random initializations (seeds 42–81). Batch-size candidates are chosen adaptively as fractions of the dataset size, rather than fixed absolute values, since a fixed batch size that is appropriate at one scale of m may be equivalent to full-batch training at another. This protocol is applied identically to the proposed VQN and to every classical baseline, so that no model is given a hyperparameter advantage.

4.2 VERIFICATION AGAINST QUANTUM CIRCUIT SIMULATION

To confirm that the closed form of Theorem 1 is an exact reformulation of the underlying quantum circuit rather than an approximation, we independently implement the circuit of Fig. 1 in PennyLane and compare its output against equation 9 evaluated directly in PyTorch, over 30 random draws of $(\mathbf{x}, \boldsymbol{\theta})$ for each $m = 2, \ldots, 6$. For every m, the maximum absolute deviation $|z_{\text{closed form}} - z_{\text{PennyLane}}|$ lies between 4.4×10^{-16} and 7.4×10^{-16}, at the level of double-precision round-off, confirming that the closed form is numerically exact. All subsequent experiments therefore train and evaluate the closed form directly, without any circuit simulation.

4.3 EXPRESSIVITY: VQN VERSUS A PARAMETER-MATCHED PERCEPTRON

We first compare a single VQN (Section 3.2) against a classical perceptron with an identical parameter count (m weights and a bias), on the m-bit parity task for $m = 2$ to 32. Figure 2 shows test accuracy as a function of m for both models, using the search protocol of Section 4.1.

For $m = 2$ through 20, the VQN achieves 100% test accuracy at every tested value of m except $m = 3$ (97.2%, still far above chance). The perceptron, in contrast, stays well below the VQN throughout, never exceeding roughly 60%, consistent with the classical result that a single linear unit cannot represent a non-linearly separable function such as parity (Minsky & Papert, 2017). Beyond $m = 20$, VQN accuracy degrades sharply, falling to 58.5% at $m = 24$, 50.3% at $m = 28$, and 50.0% (chance level) at $m = 32$, where the perceptron is also at chance. Because $m > 20$ is also where we switch from full enumeration to sampled training sets (Section 4.1), this degradation may reflect optimization difficulty, incomplete coverage of the input space by the training set, or both; we do not disentangle these factors here.

4.4 PARAMETER EFFICIENCY AGAINST CAPACITY-SCALED CLASSICAL NETWORKS

Because a perceptron cannot represent parity at all, the comparison above establishes an expressivity gap but does not indicate how much classical capacity is needed to match the VQN. To quantify this, we compare against a standard one-hidden-layer classical network with ReLU activation, sweeping its hidden width p from 1 upward and applying the same search protocol at each value of p. We select

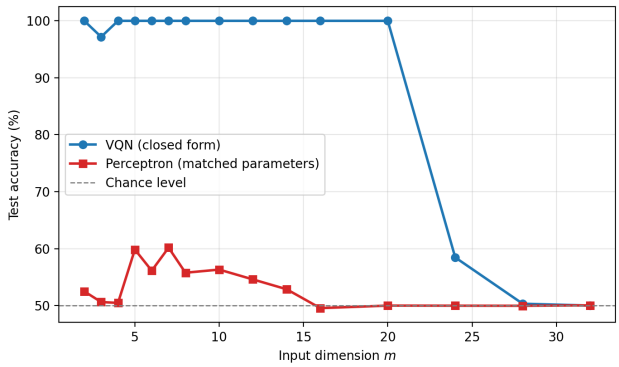


Figure 2: Test accuracy of the VQN and a parameter-matched perceptron on m-bit parity, for $m = 2$ to 32. Both models use $m + 1$ parameters at each m.

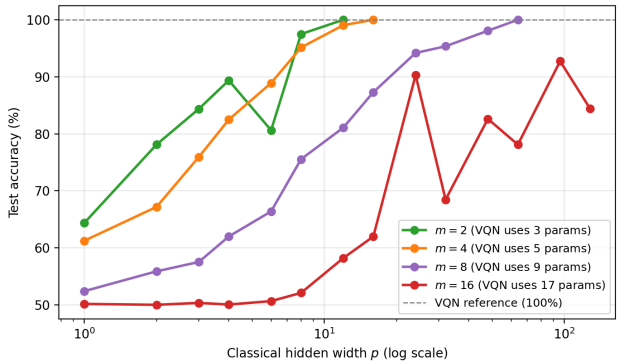


Figure 3: Test accuracy of a classical one-hidden-layer ReLU network as a function of hidden width p (log scale), for $m = 2, 4, 8, 16$. The VQN achieves 100% accuracy at each of these values of m using only $m + 1$ parameters. At $m = 16$ (red), accuracy is non-monotonic in p; see discussion below.

four representative values, $m = 2, 4, 8, 16$, at each of which the VQN achieves 100% accuracy. Figure 3 shows test accuracy as a function of classical hidden width.

Table 2 summarizes, for each m, the smallest classical network (by total parameter count) that matches the VQN's accuracy, found by sweeping p until classical accuracy reaches 99.5% or higher.

The ratio of classical to VQN parameters grows with m, from about $16\times$ at $m = 2$ to more than $135\times$ at $m = 16$, indicating that over this range the VQN's parameter-efficiency advantage widens with task scale. At $m = 16$, the classical network does not reach 99.5% accuracy for any hidden width we tested ($p \leq 128$); Table 2 therefore reports the largest tested configuration as a lower bound. At this scale, test accuracy is also non-monotonic in p (Fig. 3, red curve), varying by more than 20 percentage points across widths from 32 to 128 without approaching 99.5%. This suggests that, at this scale, the selected hyperparameter configuration sometimes trains poorly despite additional capacity, so that the optimization landscape—not merely representational capacity—becomes a limiting factor for the classical network.

As a further check that this gap is not an artifact of restricting the opponent to a single hidden layer, we additionally evaluate wider (up to $4m$ hidden units) and deeper (two-hidden-layer) classical networks at $m = 2, 4, 8, 16, 32$, using the same search protocol. Table 3 reports the results. At $m \leq 4$, moderately wide single-layer networks (width 8–16) already reach 95–100% accuracy, though still using an order of magnitude more parameters than the VQN. At $m = 8$, a two-hidden-layer network of width 16 (433 parameters) reaches 98.9% and a single-layer network of width 32 (321 parameters) reaches 95.4%; neither reaches the VQN's 100% at 9 parameters, and matching

Table 2: Smallest one-hidden-layer ReLU network whose accuracy reaches 99.5%, compared with the VQN's $m + 1$ parameters. At $m = 16$, no tested width ($p \leq 128$) reaches this level, so the entries are lower bounds.

m	VQN params	Matching p	Classical params	Ratio
2	3	12	49	$16.3\times$
4	5	16	97	$19.4\times$
8	9	64	641	$71.2\times$
16	17	> 128	> 2305	$> 135.6\times$

Table 3: Wider and deeper classical opponents. Depth 1 denotes a single hidden layer; depth 2 denotes two hidden layers of the given width.

m	Width p	Depth	Params	Test acc. (%)
2	4	1	17	89.38 ± 14.67
2	4	2	37	88.75 ± 16.72
2	8	1	33	97.50 ± 9.35
4	8	1	49	95.16 ± 5.67
4	8	2	121	98.75 ± 2.86
4	16	1	97	100.00 ± 0.00
8	16	1	161	87.30 ± 8.03
8	16	2	433	98.93 ± 1.57
8	32	1	321	95.36 ± 8.10
16	32	1	577	68.47 ± 16.06
16	32	2	1633	80.57 ± 19.71
16	64	1	1153	78.11 ± 18.35
32	64	1	2177	50.02 ± 0.15
32	64	2	6337	49.98 ± 0.14
32	128	1	4353	50.00 ± 0.15

it with a single hidden layer requires 641 parameters (Table 2). At $m = 16$, the best configuration in Table 3 (width 32, two layers, 1633 parameters) reaches only 80.6%, and the single-layer sweep of Fig. 3 peaks at roughly 93%; no classical network we tested approaches the VQN's 100% at 17 parameters. At $m = 32$, every classical configuration we tested, including the largest (6337 parameters), remains at chance level, as does the VQN (Section 4.3); at this scale neither model class learns the task, so the comparison is uninformative.

5 Conclusion and Future Work

We have shown that a single variational quantum neuron admits an exact closed-form input–output mapping (Theorem 1), verified this equivalence against direct circuit simulation to floating-point precision, and demonstrated that the neuron can be trained without any circuit simulation. On the m-bit parity task, the VQN matches or exceeds classical networks using one to two orders of magnitude more parameters for m up to 20, although its accuracy degrades sharply beyond this range. We therefore position the VQN not as a claim of general quantum advantage, but as a resource-efficient learning primitive whose behavior can be analyzed in closed form, much as a classical neuron can.

Several directions remain open. Since exact solutions exist for every m (Section 3.4), the observed transition may stem from random initialization rather than the model class itself, motivating initialization schemes informed by this structure; our variance analysis suggests this matters even more for continuous-valued inputs, where gradients concentrate at a fixed initialization scale (Appendix A.11). Extending the evaluation beyond parity, particularly to such continuous inputs, and stacking VQNs into multi-layer networks while determining whether the closed form remains tractable under composition, would clarify how far these findings generalize. Finally, deploying trained parameters on quantum hardware, where finite-shot sampling and noise come into play, would validate this primitive beyond simulation.

AI USE STATEMENT

In this work, we used a generative AI tool, Claude Opus 5.5 (Anthropic), for writing polish: correcting grammar and spelling, improving the clarity and fluency of sentences, and making wording, terminology, and punctuation consistent across sections of the manuscript. The technical content, including the method, derivations, experiments, results, and conclusions, was written by the authors, and the AI tool was used only to refine how it is expressed.

We have not used generative AI tools for generating research ideas, designing or running experiments, analyzing data, producing results or figures, or retrieving and summarizing related work; these categories are not applicable to this work.

We have reviewed all AI-assisted work. Every suggested edit was checked by the authors against the original meaning, and edits that changed technical content were rejected or corrected. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

MV Altaisky. Quantum neural network. *arXiv preprint quant-ph/0107012*, 2001.

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

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

Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Shahnawaz Ahmed, Vishnu Ajith, M Sohaib Alam, Guillermo Alonso-Linaje, Bharath AkashNarayanan, Ali Asadi, et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*, 2018.

Yudong Cao, Gian Giacomo Guerreschi, and Alán Aspuru-Guzik. Quantum neuron: an elementary building block for machine learning on quantum computers. *arXiv preprint arXiv:1711.11240*, 2017.

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

Marco Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nature communications*, 12(1): 1791, 2021b.

Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002*, 2018.

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

Zoë Holmes, Kunal Sharma, Marco Cerezo, and Patrick J Coles. Connecting ansatz expressibility to gradient magnitudes and barren plateaus. *PRX quantum*, 3(1):010313, 2022.

Tianlong Huang and Zhiyuan Li. Provable benefits of sinusoidal activation for modular addition. *arXiv preprint arXiv:2511.23443*, 2025.

Subhash C Kak. Quantum neural computing. *Advances in imaging and electron physics*, 94:259–313, 1995.

Chittaranjan Mallick, Sourav Kumar Bhoi, Sanjaya Kumar Panda, and Kalyan Kumar Jena. An efficient learning algorithm for periodic perceptron to test xor function and parity problem. *SN Applied Sciences*, 2(2):160, 2020.

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

Marvin Minsky and Seymour A Papert. *Perceptrons: An introduction to computational geometry*. the MIT Press, 2017.

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

MS Moreira, Gian Giacomo Guerreschi, Wouter Vlothuizen, Jorge F Marques, Jeroen van Straten, Shavindra P Premaratne, Xiang Zou, Hany Ali, Nandini Muthusubramanian, Christos Zachariadis, et al. Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor. *npj Quantum Information*, 9(1):118, 2023.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

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

Franz J Schreiber, Jens Eisert, and Johannes Jakob Meyer. Classical surrogates for quantum learning models. *Physical Review Letters*, 131(10):100803, 2023.

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

Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. The quest for a quantum neural network. *Quantum Information Processing*, 13(11):2567–2586, 2014.

Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331, 2019.

Maria Schuld, Alex Bocharov, Krysta M Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. *Physical Review A*, 101(3):032308, 2020.

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

Francesco Tacchino, Chiara Macchiavello, Dario Gerace, and Daniele Bajoni. An artificial neuron implemented on an actual quantum processor. *npj Quantum Information*, 5(1):26, 2019.

David Wierichs, Josh Izaac, Cody Wang, and Cedric Yen-Yu Lin. General parameter-shift rules for quantum gradients. *Quantum*, 6:677, 2022.

Table A.1: Stage-by-stage derivation of the VQN readout.

Stage	State / Operator
Init	$\lvert\psi_0\rangle = \lvert 0\rangle^{\otimes m} \otimes \lvert 0\rangle_r$
U_{enc}	$\lvert\psi_{\text{enc}}\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x}) \lvert\mathbf{s}\rangle \otimes \lvert 1\rangle_r$
H_r (1st)	$\lvert\psi_1\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x}) \lvert\mathbf{s}\rangle \otimes \lvert -\rangle_r$
$\mathrm{CR}_z + R_z$ block	$\lvert\psi_2\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x}) \lvert\mathbf{s}\rangle \otimes \frac{1}{\sqrt{2}}\big[e^{-i\theta_r(\mathbf{s})/2}\lvert 0\rangle_r - e^{i\theta_r(\mathbf{s})/2}\lvert 1\rangle_r\big]$
H_r (2nd)	$\lvert\psi_{\text{out}}\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x}) \lvert\mathbf{s}\rangle \otimes \big[-i\sin(\theta_r(\mathbf{s})/2)\lvert 0\rangle_r + \cos(\theta_r(\mathbf{s})/2)\lvert 1\rangle_r\big]$
Measure Z_r	$z(\mathbf{x}, \boldsymbol{\theta}) = -\sum_{\mathbf{s}} p_{\mathbf{s}}(\mathbf{x}) \cdot \cos\big(\theta_r(\mathbf{s})\big)$

A Closed-Form Derivation and Properties of the VQN

This appendix provides a complete step-by-step derivation of the closed-form input–output mapping of the proposed VQN (Theorem 1, Section 3.2). The derivation makes every operator, tensor factor, and phase accounting explicit, so that the mapping can be checked by hand; we additionally verify it numerically against a PennyLane circuit simulation in Section 4.2. All qubits are indexed as follows: input wires $j \in \{1, \dots, m\}$ and readout wire $r = m+1$. Unless stated otherwise, all tensor products are ordered as (wire 1) $\otimes \cdots \otimes$ (wire m) $\otimes$ (wire r). Table A.1 shows the compact derivation trace.

A.1 Setup and notation

Let $\mathbf{x} \equiv (x_1, \dots, x_m)$ be a classical feature vector with $x_j \in [0, 1]$ (continuous features are pre-normalized by equation 3 of the main text), and let the trainable parameters be $\boldsymbol{\theta} \equiv \{\theta_1, \dots, \theta_m, \theta_{m+1}\} \in \mathbb{R}^{m+1}$, where θ_j ($j = 1, \dots, m$) is the weight associated with feature x_j and θ_{m+1} is the bias. The gate $R_y(\pi x_j)$ acts on wire j as

$$R_y(\pi x_j)|0\rangle = \cos(\pi x_j/2)|0\rangle + \sin(\pi x_j/2)|1\rangle. \tag{A.1}$$

For any bitstring $\mathbf{s} \equiv (s_1, \dots, s_m) \in \{0, 1\}^m$, define the (real, non-negative) path amplitude

$$q_{\mathbf{s}}(\mathbf{x}) \equiv \prod_{j=1}^{m} \Big[\cos(\pi x_j/2)^{1-s_j} \cdot \sin(\pi x_j/2)^{s_j}\Big], \tag{A.2}$$

and the branch phase accumulated on the readout by the CR_z block and the bias rotation

$$\theta_r(\mathbf{s}) = \sum_{j=1}^{m} s_j\theta_j + \theta_{m+1}. \tag{A.3}$$

Note that

$$\sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x})^2 = \prod_j \big(\cos^2(\pi x_j/2) + \sin^2(\pi x_j/2)\big) = 1,$$

so the encoded state is normalized.

A.2 The VQN circuit as a product of unitaries

Substituting $U_{\text{ansatz}}(\boldsymbol{\theta})$ into equation 6 of the main text, the full VQN unitary reads, from right to left,

$$U_{\text{VQN}}(\mathbf{x}, \boldsymbol{\theta}) = H_r\, R_z(\theta_{m+1}) \Big(\prod_{j=1}^{m} \mathrm{CR}_z^{(j,r)}(\theta_j)\Big) H_r\, U_{\text{enc}}(\mathbf{x}), \tag{A.4}$$

where $U_{\text{enc}}(\mathbf{x}) = \bigotimes_{j=1}^{m} R_y(\pi x_j) \otimes X_r$, and $\mathrm{CR}_z^{(j,r)}(\theta_j)$ only applies $R_z(\theta_j)$ to the target (readout) qubit r when the control qubit j is in $|1\rangle$.

Remark A.1 (Order-independence of the CR_z block)**.** All gates in the product $\prod_{j=1}^{m} \mathrm{CR}_z^{(j,r)}(\theta_j)$ act as diagonal operators in the computational basis. Consequently, they mutually commute:

$$\big[\mathrm{CR}_z^{(j,r)}(\theta_j),\, \mathrm{CR}_z^{(k,r)}(\theta_k)\big] = 0, \quad \forall\, j, k \in \{1, \dots, m\}. \tag{A.5}$$

Hence the product is well defined without specifying an ordering, and the gates may be applied in any order. Commutativity does not, however, imply that the gates can be executed simultaneously: every gate acts on the readout qubit r, so in the standard two-qubit gate model they are applied sequentially, and the block has two-qubit gate depth $O(m)$ even on all-to-all connected hardware. On hardware with restricted connectivity (e.g., planar or heavy-hex superconducting layouts), additional routing operations are required to bring each input qubit into interaction with r.

A.3 Step 1: Encoded state after $U_{\mathrm{enc}}(\mathbf{x})$

Starting from $|0\rangle^{\otimes(m+1)}$, applying $U_{\mathrm{enc}}(\mathbf{x})$ yields

$$|\psi_{\mathrm{enc}}\rangle = \Big(\bigotimes_{j=1}^{m} \big(\cos(\pi x_j/2)|0\rangle + \sin(\pi x_j/2)|1\rangle\big)\Big) \otimes |1\rangle_r. \tag{A.6}$$

Expanding the tensor product in the computational basis:

$$|\psi_{\mathrm{enc}}\rangle = \sum_{\mathbf{s}\in\{0,1\}^m} q_{\mathbf{s}}(\mathbf{x})\, |s_1 \cdots s_m\rangle_{1,\ldots,m} \otimes |1\rangle_r. \tag{A.7}$$

A.4 Step 2: Interference basis after the first Hadamard on r

Applying H_r to the readout, and using $H|1\rangle = (|0\rangle - |1\rangle)/\sqrt{2} \equiv |-\rangle$:

$$|\psi_1\rangle = H_r|\psi_{\mathrm{enc}}\rangle = \sum_{\mathbf{s}\in\{0,1\}^m} q_{\mathbf{s}}(\mathbf{x})\, |s_1 \cdots s_m\rangle_{1,\ldots,m} \otimes |-\rangle_r. \tag{A.8}$$

This is the key structural observation: after the first Hadamard, every branch $\mathbf{s}$ carries the readout in the $|-\rangle$ eigenstate of Pauli-X, an equal-weight superposition of $|0\rangle_r$ and $|1\rangle_r$ with opposite signs. Any subsequent diagonal gate on r therefore imprints its rotation as a relative phase between the $|0\rangle_r$ and $|1\rangle_r$ components, never as an amplitude change on the input register. This decoupling is what enables the closed-form derivation below.

A.5 Step 3: Branch-conditioned phase accumulation

The CR_z gate $\mathrm{CR}_z^{(j,r)}(\theta_j)$ is diagonal in the computational basis of wires (j, r):

$$\mathrm{CR}_z^{(j,r)}(\theta_j)\, |s_j\rangle_j \otimes |b\rangle_r = \exp\big(-i\, s_j \theta_j (-1)^b/2\big)\, |s_j\rangle_j \otimes |b\rangle_r, \quad b \in \{0,1\}. \tag{A.9}$$

Applying the whole product to each branch of equation A.8 and then $R_z(\theta_{m+1})$ on r,

$$\Big[R_z(\theta_{m+1}) \prod_{j=1}^{m} \mathrm{CR}_z^{(j,r)}(\theta_j)\Big] |\mathbf{s}\rangle \otimes |b\rangle_r = \exp\big(-i(-1)^b\, \theta_r(\mathbf{s})/2\big)\, |\mathbf{s}\rangle \otimes |b\rangle_r, \tag{A.10}$$

with $\theta_r(\mathbf{s})$ defined in equation A.3. Applied to $|\psi_1\rangle$:

$$|\psi_2\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x})\, |\mathbf{s}\rangle \otimes \frac{1}{\sqrt{2}}\Big[e^{-i\theta_r(\mathbf{s})/2}|0\rangle_r - e^{i\theta_r(\mathbf{s})/2}|1\rangle_r\Big]. \tag{A.11}$$

A.6 Step 4: Second Hadamard and phase-to-population conversion

Applying H_r once more and using

$$H_r\, \frac{1}{\sqrt{2}}\Big(e^{-i\theta/2}|0\rangle - e^{i\theta/2}|1\rangle\Big) = -i\sin(\theta/2)|0\rangle + \cos(\theta/2)|1\rangle, \tag{A.12}$$

we obtain the pre-measurement state

$$|\psi_{\mathrm{out}}(\mathbf{x}, \boldsymbol{\theta})\rangle = \sum_{\mathbf{s}} q_{\mathbf{s}}(\mathbf{x})\, |\mathbf{s}\rangle \otimes \Big[-i\sin\big(\theta_r(\mathbf{s})/2\big)|0\rangle_r + \cos\big(\theta_r(\mathbf{s})/2\big)|1\rangle_r\Big]. \tag{A.13}$$

A.7 Step 5: Measurement of Z_r and the closed-form response

The observable is the Pauli-Z on the readout wire (with I on all other wires). The expectation value decomposes into a sum over branches because different $\mathbf{s}$ label orthogonal input-register basis states:

$$\langle Z_r \rangle_{\mathbf{x},\boldsymbol{\theta}} = \langle \psi_{\text{out}} | (I \otimes Z_r) | \psi_{\text{out}} \rangle = \sum_{\mathbf{s}} p_{\mathbf{s}}(\mathbf{x}) \cdot \langle \phi_{\mathbf{s}} | Z_r | \phi_{\mathbf{s}} \rangle, \tag{A.14}$$

where $p_{\mathbf{s}}(\mathbf{x}) \equiv q_{\mathbf{s}}(\mathbf{x})^2$ and $|\phi_{\mathbf{s}}\rangle \equiv -i \sin(\theta_r(\mathbf{s})/2)|0\rangle + \cos(\theta_r(\mathbf{s})/2)|1\rangle$. A direct computation gives

$$\langle \phi_{\mathbf{s}} | Z_r | \phi_{\mathbf{s}} \rangle = \sin^2\big(\theta_r(\mathbf{s})/2\big) - \cos^2\big(\theta_r(\mathbf{s})/2\big) = -\cos\big(\theta_r(\mathbf{s})\big). \tag{A.15}$$

Substituting equation A.15 back into equation A.14:

$$\langle Z_r \rangle_{\mathbf{x},\boldsymbol{\theta}} = -\sum_{\mathbf{s}\in\{0,1\}^m} p_{\mathbf{s}}(\mathbf{x}) \cdot \cos\Big(\sum_{j=1}^{m} s_j \theta_j + \theta_{m+1}\Big). \tag{A.16}$$

Equation A.16 is the closed-form scalar response of the VQN, i.e., the readout $z(\mathbf{x}, \boldsymbol{\theta})$ of equation 9 in the main text.

A.8 Theorem 1, restated

Theorem 1 (restated). *Let $U_{\text{VQN}}(\mathbf{x}, \boldsymbol{\theta})$ be defined by equation A.4, and let $q_{\mathbf{s}}(\mathbf{x})$ and $\theta_r(\mathbf{s})$ be defined by equation A.2 and equation A.3. Then the scalar readout of the VQN admits the exact closed form*

$$z(\mathbf{x}, \boldsymbol{\theta}) \equiv \langle 0|^{\otimes(m+1)} U_{\text{VQN}}^{\dagger}(\mathbf{x}, \boldsymbol{\theta})\, Z_r\, U_{\text{VQN}}(\mathbf{x}, \boldsymbol{\theta}) |0\rangle^{\otimes(m+1)} = -\sum_{\mathbf{s}} p_{\mathbf{s}}(\mathbf{x}) \cdot \cos\big(\theta_r(\mathbf{s})\big). \tag{A.17}$$

Proof. Combine Steps 1–5 (Sections A.3–A.7 above). The state after $U_{\text{VQN}}(\mathbf{x}, \boldsymbol{\theta})$ is $|\psi_{\text{out}}\rangle$ of equation A.13; the observable is diagonal in the branch basis $\{|\mathbf{s}\rangle\}$ on the input register, giving equation A.14; the readout expectation on each branch is $-\cos(\theta_r(\mathbf{s}))$ by equation A.15; summing over branches gives equation A.17. □

A.9 Corollaries and immediate consequences

A.9.1 Bounded, nonlinear, periodic activation

From equation A.17, $z(\mathbf{x}, \boldsymbol{\theta}) \in [-1, +1]$ because it is, up to a sign, a convex combination of cosines (the weights $p_{\mathbf{s}}(\mathbf{x})$ are non-negative and sum to 1). The mapping is (i) bounded, (ii) 2π-periodic in each θ_j, and (iii) nonlinear in both $\mathbf{x}$ and $\boldsymbol{\theta}$ without any classical activation function, the nonlinearity being entirely supplied by the interferometric readout.

A.9.2 Fourier structure

Since $a_j \equiv \sin^2(\pi x_j/2) = \big(1 - \cos(\pi x_j)\big)/2$, each factor of equation 10 is affine in $\cos(\pi x_j)$, so $z(\mathbf{x}, \boldsymbol{\theta})$ is a real trigonometric polynomial in $\mathbf{x}$ whose accessible frequency vectors are $\{-1, 0, 1\}^m$ in units of π. This is exactly the spectrum that Schuld et al. (2021) predict for a single Pauli-rotation encoding per feature: the VQN accesses 3^m frequency vectors in the inputs without any repeated data uploading. With respect to the parameters, writing $\cos(\theta_r(\mathbf{s})) = \big(e^{i\theta_r(\mathbf{s})} + e^{-i\theta_r(\mathbf{s})}\big)/2$ shows that z is a trigonometric polynomial in $\boldsymbol{\theta}$ whose frequency vectors are $\pm(\mathbf{s}, 1)$ for $\mathbf{s} \in \{0, 1\}^m$. This periodic structure is the basis for our comparison to classical periodic-activation neurons in Section 2.2.

A.9.3 Classical simulability and its interpretation

Although equation A.17 is written as a sum over 2^m branches, it factorizes over features. With $a_j \equiv \sin^2(\pi x_j/2)$, the path probability is $p_{\mathbf{s}}(\mathbf{x}) = \prod_j (1 - a_j)^{1-s_j} a_j^{s_j}$ and the branch phase satisfies $e^{i\theta_r(\mathbf{s})} = e^{i\theta_{m+1}} \prod_j e^{i s_j \theta_j}$. Applying the distributive identity $\sum_{\mathbf{s}} \prod_j f_j(s_j) =$

$\prod_j \sum_{s_j \in \{0,1\}} f_j(s_j)$ then yields equation 10. Consequently, equation A.17 can be evaluated classically in $O(m)$ arithmetic operations for any input, and the VQN does not constitute a quantum computational advantage in the complexity-theoretic sense. This is by design: the closed form is precisely what allows us to bypass quantum circuit simulation entirely and train the neuron directly in a standard automatic-differentiation framework, which is the approach taken throughout our experiments (Section 4).

The role of the quantum circuit is therefore physical realizability rather than computational speedup: the same function is implemented natively with $m+1$ qubits and m two-qubit gates, and since its gradients coincide with the parameter-shift rule (Appendix A.10), parameters trained classically through the closed form transfer directly to hardware. Equation 10 also admits a simple probabilistic reading: $z(\mathbf{x}, \boldsymbol{\theta}) = -\mathbb{E}\big[\cos\big(\sum_j S_j \theta_j + \theta_{m+1}\big)\big]$ with independent $S_j \sim \mathrm{Bernoulli}(a_j)$, i.e., a cosine neuron applied to a stochastically binarized input and averaged over the randomness. For binary inputs, this randomness vanishes and the expression reduces to equation 11.

A.10 Analytic gradients (parameter-shift rule compatibility)

Because each θ_j enters $z(\mathbf{x}, \boldsymbol{\theta})$ only through cosines of the form $\cos(s_j \theta_j + c)$ with $s_j \in \{0, 1\}$, the partial derivatives admit closed forms that coincide with the parameter-shift rule of Mitarai et al. (2018) and Schuld et al. (2019):

$$\frac{\partial z}{\partial \theta_j} = \sum_{\mathbf{s}} s_j \, p_{\mathbf{s}}(\mathbf{x}) \cdot \sin\big(\theta_r(\mathbf{s})\big) = \frac{z\big(\mathbf{x}, \boldsymbol{\theta} + \frac{\pi}{2}\mathbf{e}_j\big) - z\big(\mathbf{x}, \boldsymbol{\theta} - \frac{\pi}{2}\mathbf{e}_j\big)}{2}, \tag{A.18}$$

$$\frac{\partial z}{\partial \theta_{m+1}} = \sum_{\mathbf{s}} p_{\mathbf{s}}(\mathbf{x}) \cdot \sin\big(\theta_r(\mathbf{s})\big) = \frac{z\big(\mathbf{x}, \boldsymbol{\theta} + \frac{\pi}{2}\mathbf{e}_{m+1}\big) - z\big(\mathbf{x}, \boldsymbol{\theta} - \frac{\pi}{2}\mathbf{e}_{m+1}\big)}{2}. \tag{A.19}$$

The two-shift rule holds exactly for every VQN parameter. While all experiments in this paper train $z(\mathbf{x}, \boldsymbol{\theta})$ by standard backpropagation through the closed form (Section 3.4), this exact agreement confirms that the gradients used for training coincide with those that the physical circuit yields via parameter shifts (in the infinite-shot limit), which is a prerequisite for eventual deployment on real quantum hardware.

A.11 Gradient variance at random initialization

Let all $m+1$ parameters be drawn i.i.d. from $\mathcal{N}(0, \gamma^2)$ and fix $\mathbf{x}$; we write γ for the initialization scale to avoid a clash with the sigmoid $\sigma(\cdot)$. Differentiating equation 10 with respect to θ_j gives

$$\frac{\partial z}{\partial \theta_j} = \mathrm{Im}\,[W_j], \qquad W_j \equiv a_j \, e^{i(\theta_j + \theta_{m+1})} \prod_{k \neq j} c_k, \qquad c_k \equiv 1 - a_k + a_k e^{i\theta_k}. \tag{A.20}$$

Since $\mathbb{E}[e^{i\theta}] = e^{-\gamma^2/2}$ is real, $\mathbb{E}[W_j]$ is real and hence $\mathbb{E}[\partial z / \partial \theta_j] = 0$. Using $(\mathrm{Im}\, W)^2 = \big(|W|^2 - \mathrm{Re}\, W^2\big)/2$ and the independence of the parameters, we obtain the exact expression

$$\mathrm{Var}_{\boldsymbol{\theta}}\left[\frac{\partial z}{\partial \theta_j}\right] = \frac{a_j^2}{2}\Big[\prod_{k \neq j} \mu_k - e^{-4\gamma^2} \prod_{k \neq j} \nu_k\Big], \tag{A.21}$$

where

$$\mu_k \equiv \mathbb{E}|c_k|^2 = 1 - 2a_k(1-a_k)\big(1 - e^{-\gamma^2/2}\big), \qquad \nu_k \equiv \mathbb{E}[c_k^2] = (1-a_k)^2 + 2a_k(1-a_k)e^{-\gamma^2/2} + a_k^2 e^{-2\gamma^2}. \tag{A.22}$$

Binary inputs. If $x_k \in \{0, 1\}$ for all k, then $a_k \in \{0, 1\}$, so $\mu_k = 1$ and $\nu_k = e^{-2\gamma^2 a_k}$. For $x_j = 1$, equation A.21 reduces to

$$\mathrm{Var}_{\boldsymbol{\theta}}\left[\frac{\partial z}{\partial \theta_j}\right] = \frac{1}{2}\Big[1 - \exp\big(-2\gamma^2(|\mathbf{x}| + 1)\big)\Big], \qquad |\mathbf{x}| \equiv \sum_k x_k, \tag{A.23}$$

which does not decay with m and in fact increases toward $1/2$ as the Hamming weight of the input grows; for $x_j = 0$ the gradient vanishes identically, since θ_j does not enter z.

Continuous inputs. If $a_k \in (0, 1)$, then $\mu_k < 1$ for any $\gamma > 0$. Since $\nu_k \geq 0$, equation A.21 gives the upper bound

$$\mathrm{Var}_{\boldsymbol{\theta}}\left[\frac{\partial z}{\partial \theta_j}\right] \leq \frac{a_j^2}{2} \prod_{k \neq j} \mu_k, \tag{A.24}$$

so at a fixed initialization scale the gradient variance decays exponentially in the number of features whose a_k is bounded away from 0 and 1. This is a barren-plateau-type concentration that arises from averaging over the 2^m encoded branches rather than from circuit depth. Because $1 - \mu_k = O(\gamma^2)$, choosing $\gamma^2 = O(1/m)$ keeps $\prod_{k \neq j} \mu_k$ bounded away from zero, so that this upper bound no longer forces exponential decay; whether such small-angle initialization also yields good training is left open.

These results describe the gradient of the readout at initialization and do not by themselves guarantee that gradient-based optimization will locate a good solution. In particular, for binary inputs, the setting of all our experiments, the readout gradient variance does not vanish, so the degradation we observe beyond $m \approx 20$ (Section 4.3) is not explained by vanishing gradients at initialization. Since exact solutions exist for every m (e.g., $\theta_j = \pi/2$ for all j with a suitable bias; Section 3.4), we conjecture that this transition reflects the difficulty of reaching such solutions from random initialization rather than a limitation of the model class itself; characterizing its origin is left for future work.