

Classical Neural Networks in Quantum Constant Depth

Abstract

How quantum computing and neural network architectures can be meaningfully integrated is a central question in quantum artificial intelligence. Current approaches mainly follow two paradigms: variational quantum circuits, and quantum algorithms for the linear-algebraic primitives of classical neural networks. Here we propose a third route, grounded in the complexity-theoretic analysis of classical neural networks and in shallow quantum circuit complexity. We show that a neuron with n inputs and m bits of precision can be implemented in a model of computation where constant-depth quantum circuits are combined with measurements, classical processing, and feedforward. The resulting quantum depth is *independent* of both n and m . Our construction relies on constant-depth implementations of arithmetic and comparison primitives, including addition, multi-number addition, sorting, one-hot encoding, and monotone activation functions. For these primitives we give explicit resource counts. The same construction can be applied coherently in superposition, enabling superpositions over inputs or over multiple neurons evaluated on the same input. It also admits a distributed evaluation protocol, does not require QRAM, and does not introduce barren plateaus. We validate the construction through a hybrid classifier for molecular pose prediction of protein–ligand complexes. We compare classical simulations with executions on a trapped-ion quantum computer. This provides a new perspective on the interplay between classical neural computation and shallow quantum computation.

Introduction

The current landscape of quantum artificial intelligence shows mainly two possible directions on how to use quantum computation to enhance neural networks. The first, called the variational paradigm, consist in parameterized quantum circuits (also known as variational circuits or quantum neural networks). While there are some promising results in this direction [Gil+25], the open question is whether we can build circuits that satisfy simultaneously three conditions: expressivity, trainability, and classical non-simulability [Lar+25; Ber+26; PK24]. The second approach uses the quantum computer to perform faster the linear algebraic operations of classical neural networks. While the speedups that can be found in this approach have provable guarantees, they rely on QRAM, and so they are considered out of the scope of current quantum hardware [Che+24; Guo+24].

Motivated by the limitations of these two approaches, we offer a third perspective. It is based on two different theoretical foundations: the complexity-theoretic analysis of classical neural networks and shallow quantum circuit complexity. We introduce a hybrid model that combines insights from both, resulting in a neuron implemented by a hybrid circuit with constant quantum depth. We implement two variants of the neuron on a 92-qubit quantum computer and test it on a molecular pose prediction task.

Complexity of classical neural networks. The relationship between Boolean circuit complexity and neural-network models has long been an active area of study [Par94]. In this sense, the class of classical constant-depth circuits TC^0 can be interpreted as a model for neural networks composed of Boolean neurons. Transformer architectures with saturated attention exhibit computational capabilities that can be as powerful as TC^0 circuits [MSS22; Str23]. Beyond constant-depth complexity classes, Li et al. [Li+24] show that chain-of-thought provides bounded-depth transformers with an external source of sequential computation, and similar ideas can be used to show that diffusion language models are optimal parallel samplers [JHC25].

Hybrid computation with shallow circuits. Quantum analogues of classical constant-depth complexity classes have been investigated extensively over the past several decades. Takahashi and Tani [TT16] *collapsed the constant-depth quantum hierarchy with FanOut*: they showed that $\text{QNC}_f^0 = \text{QAC}_f^0 = \text{QTC}_f^0$, exhibiting a polynomial size constant-depth circuit that — using FanOut gates — pulls threshold gates inside QNC_f^0 . A complementary way to extend shallow quantum circuits is to allow classical feedforward between quantum layers. This idea was formalized in [Buh+24] as the LAQCC complexity class, or LAQCC model of computation. An LAQCC computation consists of a constant number of adaptive rounds: in each round, one applies a constant-depth quantum circuit (QNC^0), measures some qubits, processes the measurement outcomes with a log-depth classical circuit (NC^1), and uses the resulting classical information to choose or control the next constant-depth quantum circuit. This model captures the regime of near-term hardware augmented with a classical co-processor. Very importantly, in this model we can implement FanOut gates. With the additional observation that $\text{TC}^0 \subset \text{QNC}_f^0 = \text{QTC}_f^0 \subseteq \text{LAQCC}$, this is the precise reason why every layer of a finite-precision classical neuron

can be lifted to a constant-depth quantum circuit with feedforward. There are other hints of the importance of the interplay between classical and quantum computation: the class of problems that can be solved by a quantum computer with a polynomial number of layers of constant-depth quantum circuits with adaptive control ($\text{BPP}^{\text{QNC}^{\text{BPP}}_{O(1)}}$ — a definition which shares many similarities with LAQCC) is not included in either BPP^{QNC} or QNC^{BPP} [Aro+23].

Main result

Theorem (Informal). *Let $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ be a Lipschitz activation function, $\vec{w} \in \mathbb{R}^n$ be the weight vector, $\vec{x} \in \mathbb{R}^n$ be the input vector. Then there exists a hybrid circuit that computes the output of a neuron $z = \sigma(\vec{w}^\top \vec{x})$ with precision m in a quantum-depth that is independent of n and m .*

There are few interesting consequences of this construction.

- **Tradeoffs.** The construction allows for various tradeoffs between depth and width in n and/or m . For example, we can implement a neuron with a depth that is constant in n but polynomial in m , and vice-versa.
- **Superposition.** The whole circuit can be applied in superposition. For example, we can create superpositions over possible inputs (e.g. the output of a previous hidden layer or a batch $\mathcal{B}$ of input data) $\sum_{i \in [\mathcal{B}]} |i\rangle |\vec{x}_i\rangle$, producing the state $\sum_{i \in [\mathcal{B}]} |i\rangle |\sigma(\vec{w}^\top \vec{x}_i)\rangle$ in constant depth. Alternatively, we can create states proportional to $\sum_{j \in [L]} |j\rangle |\sigma(\vec{w}_j^\top \vec{x})\rangle$ representing a superposition of multiple neurons evaluated on the same input x . These states can then serve as inputs to three natural downstream architectures: further layers of the hybrid LAQCC model developed here, variational quantum circuits, or quantum-linear-algebra-based neural networks.
- **No barren plateaus and no QRAM.** Since we are able to implement exactly the classical neuron, we don't suffer from the barren plateaus problem. Trainable parameters and input data enter the circuit through controlled phase rotation, without the need for QRAM.
- **Distributed and private neuron evaluation.** The construction also has a natural distributed interpretation. Suppose that n parties each hold one coordinate of the input vector $\vec{x}$. If the parties are given one qubit each from a shared quantum state, then each party can apply an operation determined by its own input coordinate. After these local operations, the parties return their qubits to a central processor, which can continue the computation on the joint state without ever learning the individual input coordinates.
- **Future directions.** Possible directions are *i)* decreasing the reliance on classical computation — for example, by exploring recent separations between QNC^0 and BPTF [Hsi+24] or catalytic computation [Buh+25] *ii)* explore implications of recent results on quantization of neural networks [Yan+25]¹.

Methodology

We begin by revisiting the constant-depth primitives used in previous works, including the OR gate, the EXACT gate, and the THRESHOLD gate [HŠ05; TT16]. Prior analyses established only asymptotic ancilla requirements and did not provide an exact accounting of the circuit depth. In contrast, we give explicit resource counts: we track the number of ancillas, the total depth, and the number of FanOut layers, and we further *optimize* these constructions when possible. We then use these optimized primitives to build the following subroutines:

- **Bitstring transformations.** We show subroutines for creating one-hot encoding from unary encoding, sorting, Balanced-Majority-of-Four², single-bit extraction of Hamming weight of a bitstring, and single-bit extraction of Hamming weight of a sum of bits.
- **Addition and multi-number addition.** We construct a quantum circuit which maps $|x\rangle_m |y\rangle_m |0\rangle \mapsto |x\rangle |x+y\rangle_{m+1}$ in depth 47, of which 19 are FanOut layers with arity at most $\lfloor (m+1)^2/4 \rfloor$. Inspired by the classical circuits in [HHK91], we show quantum version of circuits for adding multiple numbers in constant depth.

¹Techniques to reduce the precision of the weights and activations of a neural network to reduce the number of bits required to store the weights and the floating point precision of the calculations.

²A generalization of majority on 2 strings with 4 symbols.

- **Quantum Fourier Transform.** Following ideas of [HŠ05] we implement the QFT with high probability. Extending [HŠ05], we give an explicit construction in our manuscript, counting the exact depth and number of gates and ancilla qubits.
- **Comparator.** We build a circuit $\text{CMP}^n(x, y)$, which returns 1 if x is greater than y and 0 otherwise, with depth 45, of which 19 FanOut layers.
- **Activation function.** We give two constant-depth decomposers, algorithms that takes as input a function $f : \{0, 1\}^m \mapsto \mathbb{R}$ and produce as output a quantum circuit that computes the function. In particular, given a discretisation of the co-domain of σ in K intervals and an n -bit input, they produce the one-hot encoding of the bin index. The first is a “generic” decomposer that can be used to decompose any (activation) function from $\mathbb{R}$ to $\mathbb{R}$. This decomposer is based on ideas from [Hsi+24]. For each interval i , the indicator $g_i(x) = \text{OR}_{j \in S_i} \text{EXACT}_j(x)$ is computed by hardwiring the equality-tests on the precomputed $S_i \subseteq \{0, 1\}^m$: the set of inputs of f that gets mapped into the i -th interval. The resulting circuit evaluates all K indicators in parallel. However, most interesting activation functions are monotone. This motivates the second decomposer: in this case, the indicator $g_i(x)$ can be built from two comparators, yielding a total depth of 93, of which 40 are layers of FanOut gates.

We also report optimized depth and space results for the implementation of a FanOut gate on the ground state, and its uncomputation, which we use in our experiments. Importantly, the combination of different subroutines can be often optimized further, resulting in a shallower depth than the sum of the depths of the subroutines.

Neuron decomposition. The neuron is decomposed into three stages.

1. Build the Fourier state $|\psi_y\rangle = \bigotimes_{k=0}^{m-1} \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i y/2^k} |1\rangle)$ encoding the pre-activation $y = \vec{w}^\top \vec{x}$ in the phase: for $k \in [m]$, each vector component x_j controls a Z rotation proportional to $w_j x_j / 2^k$. The sums of the products $w_j x_j$ across all n inputs are accumulated into a single qubit via a “phase-kickback” implemented via two FanOut layers [HŠ05; All+24].
2. Apply the inverse QFT to write the binary value of y on an m -qubit register $|y\rangle$. We optimize the constant-depth implementation sketched in [HŠ05]. The QFT-inverse converts this phase-encoded state into an m -bit estimate of the inner product, giving a *constant-depth subroutine for inner-product computation*.
3. Apply a constant-depth activation oracle $|y\rangle_m |0\rangle \mapsto |y\rangle_m |\sigma(y)\rangle$, generalising the Boolean-function-evaluation framework of [All+24] from Boolean targets to K -level quantised activations and from a single function to the whole family of bounded Lipschitz σ . Each stage relies on new constant-depth arithmetic circuits for which we track *exact* gate counts (depth, fan-out layers, ancilla qubits).

Experiments

We use the CSAR molecular pose-classification benchmark of [Rag+17], through the PyTorch reimplementaion of [MM22], as a testbed for the proposed quantum neuron. The input is a voxelized receptor–ligand grid $x \in \mathbb{R}^{C \times D \times H \times W}$, with $C = 28$ atom-type channels and $D = H = W = 48$. We first perform a validation where we show that width can compensate for reducing depth, to exploit the tradeoffs between depth and width introduced by our neuron architecture. We show that a network of 3 convolutional layers with 512 neurons in the last feed forward layer can achieve performances comparable to a network of 5 convolutional layers with 64 neurons in the last feed-forward layer. For the quantum experiment, we reduced the last hidden layer to 4 neurons (hence $n = 5$ accounting for the bias of the neuron). In one experiment we use $m = 3$ bits of precision, and we use QFT^{-1} of depth 9. As nonlinearity, we implement a sigmoid function with another circuit of depth 9. We compare noiseless simulation, depolarizing-noise simulation, and a hardware-inspired simulation. To remove the randomization error of the quantum algorithms, and to isolate the quantum noise we introduce a per-sample adjustment of the bias of a neuron that snaps the neuron’s preactivation to the nearest value representable by m bits of precision. Current results show that the quantum neuron (executed on real hardware and in classical simulations) is able to match the performance of the classical network. Finally, we also run the same pipeline on a larger final neuron with 32 hidden inputs plus a bias, using $m = 1$ bit of precision.

References

- [All+24] Jonathan Allcock et al. “Constant-depth circuits for Boolean functions and quantum memory devices using multi-qubit gates”. In: *Quantum* 8 (2024), p. 1530.
- [Aro+23] Atul Singh Arora et al. “Quantum depth in the random oracle model”. In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 1111–1124.
- [Ber+26] Pablo Bermejo et al. “Quantum convolutional neural networks are effectively classically simulable”. In: *Prx Quantum* 7.2 (2026), p. 020304.
- [Buh+24] Harry Buhrman, Marten Folkertsma, Bruno Loff, and Niels MP Neumann. “State preparation by shallow circuits using feed forward”. In: *Quantum* 8 (2024), p. 1552.
- [Buh+25] Harry Buhrman et al. “Quantum catalytic space”. In: *arXiv preprint arXiv:2506.16324* (2025).
- [Che+24] El Amine Cherrat et al. “Quantum vision transformers”. In: *Quantum* 8.arXiv: 2209.08167 (2024), p. 1265.
- [Gil+25] Elies Gil-Fuster, Casper Gyurik, Adrián Pérez-Salinas, and Vedran Dunjko. “On the relation between trainability and dequantization of variational quantum learning models”. In: *International Conference on Learning Representations*. Vol. 2025. 2025, pp. 24069–24093.
- [Guo+24] Naixu Guo et al. “Quantum linear algebra is all you need for transformer architectures”. In: *arXiv e-prints* (2024), arXiv–2402.
- [HHK91] Thomas Hofmeister, Walter Hohberg, and Susanne Köhling. “Some notes on threshold circuits, and multiplication in depth 4”. In: *International Symposium on Fundamentals of Computation Theory*. Springer. 1991, pp. 230–239.
- [HŠ05] Peter Høyer and Robert Špalek. “Quantum fan-out is powerful”. In: *Theory of computing* 1.1 (2005), pp. 81–103.
- [Hsi+24] Min-Hsiu Hsieh, Leandro Mendes, Michael de Oliveira, and Sathyawageeswar Subramanian. “Unconditionally separating noisy QNC0 from bounded polynomial threshold circuits of constant depth”. In: *arXiv preprint arXiv:2408.16378* (2024).
- [JHC25] Haozhe Jiang, Nika Haghtalab, and Lijie Chen. “Diffusion Language Models are Provably Optimal Parallel Samplers”. In: *arXiv preprint arXiv:2512.25014* (2025).
- [Lar+25] Martin Larocca et al. “Barren plateaus in variational quantum computing”. In: *Nature Reviews Physics* 7.4 (2025), pp. 174–189.
- [Li+24] Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. “Chain of thought empowers transformers to solve inherently serial problems”. In: *The Twelfth International Conference on Learning Representations*. 2024.
- [MM22] Rocco Meli and Andrew McNutt. *gninatorch*. Version 0.0.2. July 2022. DOI: 10.5281/zenodo.6943066. URL: <https://github.com/RMeli/gnina-torch>.
- [MSS22] William Merrill, Ashish Sabharwal, and Noah A Smith. “Saturated transformers are constant-depth threshold circuits”. In: *Transactions of the Association for Computational Linguistics* 10 (2022), pp. 843–856.
- [Par94] Ian Parberry. *Circuit complexity and neural networks*. MIT press, 1994.
- [PK24] Chae-Yeun Park and Nathan Killoran. “Hamiltonian variational ansatz without barren plateaus”. In: *Quantum* 8 (2024), p. 1239.
- [Rag+17] Matthew Ragoza et al. “Protein–ligand scoring with convolutional neural networks”. In: *Journal of chemical information and modeling* 57.4 (2017), pp. 942–957.
- [Str23] Lena Strobl. “Average-hard attention transformers are constant-depth uniform threshold circuits”. In: *arXiv preprint arXiv:2308.03212* (2023).
- [TT16] Yasuhiro Takahashi and Seiichiro Tani. “Collapse of the hierarchy of constant-depth exact quantum circuits”. In: *computational complexity* 25.4 (2016), pp. 849–881.
- [Yan+25] Jiawei Yang, Zhongbo Li, Zeqin Feng, and Yongqiang Xie. “A survey on neural network quantization”. In: *Proceedings of the 2025 6th International Conference on Computer Information and Big Data Applications*. 2025, pp. 384–394.