

Quantum-Enhanced Large Language Models on Quantum Hardware via Cayley Unitary Adapters

Borja Aizpurua^{1,2}, Sukhbinder Singh³, Augustine Kshetrimayum¹, Saeed S. Jahromi^{1,4}, and Román Orús^{1,4,5}

¹Multiverse Computing, Paseo de Miramón 170, 20014 San Sebastián, Spain

²Tecnun, University of Navarra, San Sebastián, Spain

³Multiverse Computing, 192 Spadina Avenue, Toronto, ON M5T 2C2, Canada

⁴Donostia International Physics Center (DIPC), San Sebastián, Spain

⁵Ikerbasque Foundation for Science, Bilbao, Spain

Abstract

We introduce *Cayley Unitary Adapters* (CUAs), a hardware-efficient way to inject trainable quantum parameters into a frozen large language model (LLM). Each adapter is a block-diagonal unitary whose blocks are parameterised by the Cayley transform of a skew-symmetric matrix; the parameters are trained *entirely classically* by knowledge distillation, and the trained two-qubit blocks are then executed on real superconducting hardware via amplitude encoding. On Llama 3.1 8B, a single adapter with $\sim 6,000$ extra parameters improves WikiText perplexity by 1.43%, with end-to-end autoregressive generation validated on a 156-qubit IBM Quantum System Two processor and shown to be robust to hardware noise. A systematic study on SmolLM2 (135M) shows perplexity improving monotonically with block dimension, a sharp noise-driven phase transition at three qubits, and an operator-entanglement analysis separating the CUA from a depth-matched variational circuit. To our knowledge this is the first quantum enhancement of a production-scale, pre-trained LLM for autoregressive generation on real gate-based hardware.

1 Introduction

Large language models have redefined generative AI [1, 2], but every trainable parameter occupies classical memory, and scaling a deployed model demands a proportional expansion of compute [3]. Quantum computing offers a different resource paradigm: an n -qubit register spans a 2^n -dimensional space while physical resources scale only linearly in n . Variational quantum circuits (VQCs) [4] are the standard route to injecting quantum parameters into ML models, typically as fixed brickwork ansätze. Prior quantum-NLP work has addressed classification [5], toy grammars [6], hybrid LLM fine-tuning [7], and quantum self-attention on a 72-qubit processor [8], but these run on simulators, target classification rather than autoregressive generation, or operate at restricted scale. We close this gap with CUAs, building on tensor-network disentangler constructions for porting classical networks to quantum devices [9, 10].

2 Cayley Unitary Adapters

A CUA replaces a selected projection W of the transformer by a quantum block acting on its input (Fig. 1). The adapter is a block-diagonal unitary (BDU) $Q = \bigoplus_{i=1}^k Q_i$ of k independent $2^n \times 2^n$ orthogonal blocks, with adapter input dimension $d = k 2^n$ matching the residual-stream dimension. Each block is parameterised by the Cayley transform [11, 12]

$$Q_i = (I - \tfrac{1}{2}K_i)(I + \tfrac{1}{2}K_i)^{-1}, \quad K_i = -K_i^\top, \quad (1)$$

with $n(n-1)/2$ free parameters per block. A 4×4 block (two qubits) needs only six parameters yet realises a full orthogonal rotation: for $d = 4,096$ (Llama 3.1 8B) a CUA holds 1,024 blocks and 6,144 parameters; for $d = 576$ (SmolLM2), 144 blocks and 864 parameters. Because quantum

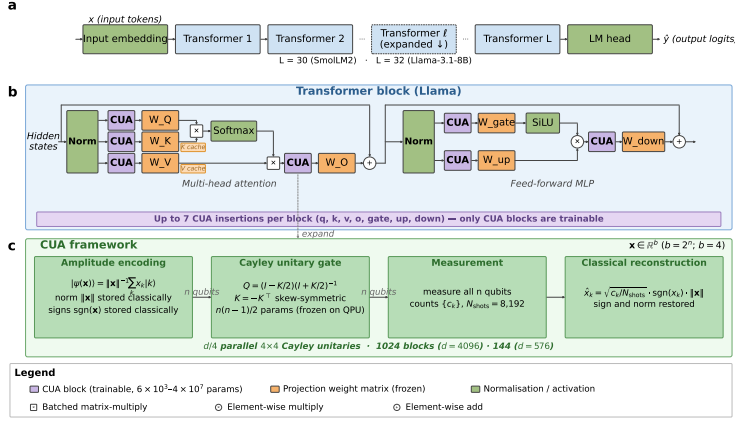


Figure 1: Cayley Unitary Adapter (CUA) architecture. (a) Frozen transformer backbone. (b) CUA blocks (purple) are inserted on the input side of each linear projection; only CUA blocks are trainable. (c) Each CUA applies $d/4$ independent 4×4 Cayley unitaries in parallel, each amplitude-encoded, transformed by a Cayley gate $Q^\top$ trained classically and frozen on the QPU, measured, and reconstructed with the classically stored sign and norm.

Table 1: Llama 3.1 8B WikiText perplexity by adapter configuration. Improvement is relative to the base model.

Configuration	PPL	Extra params	Impr.
Base (no adapter)	8.877	—	—
2-qubit BDU (1 layer)	8.752	6,144	+1.43%
+ Heron r2 noise	8.759	6,144	+1.34%
Full unitary + sign (1 layer)	8.618	8.38M	+2.92%
Unconstrained (1 layer)	8.585	16.77M	+3.29%
2-qubit BDU (192 sublayers)	8.393	—	+5.45%

measurements return non-negative probabilities, the adapted layer applies an input-dependent sign correction, $\mathbf{y} = W(|Q(\mathbf{x})| \odot \text{sgn}(\mathbf{x}))$. All backbone weights stay frozen; only the K_i are trained, by supervised fine-tuning combined with online knowledge distillation [13] from the original model.

Execution on hardware. After training, each 4-vector slice $\mathbf{x}$ is amplitude-encoded as $|\psi\rangle = \|\mathbf{x}\|^{-1} \sum_{k=0}^3 x_k |k\rangle$, the gate $Q^\top$ is applied, and both qubits are measured over $N_{\text{shots}} = 8,192$ shots; the output is reconstructed as $\hat{y}_k = \sqrt{c_k/N_{\text{shots}}} \text{sgn}(x_k) \|\mathbf{x}\|$. A generic $d \times d$ unitary would need exponential synthesis depth; the BDU instead factorises into independent two-qubit blocks run in parallel. Greedy maximum-matching packs up to 64 disjoint two-qubit lanes per wide circuit, so all 1,024 Llama blocks per token fit in 16 packed circuits (Fig. 2).

3 Results

Primary result: Llama 3.1 8B. We insert a single CUA into the value projection ($\mathbf{v}_{\text{proj}}$) of the 7th attention layer of the unmodified model. WikiText perplexity (PPL) improves from 8.877 to 8.752 (+1.43%) with 6,144 parameters (Table 1). Dense input-dimension unitaries reach 8.618 and unconstrained matrices 8.585, so the two-qubit BDU captures roughly half the upper-bound gain at $\sim 2,730 \times$ fewer parameters. Noise simulation under IBM Heron r2 error rates barely perturbs the result (8.759 vs. 8.752, 0.08%). Extending to 192 sublayers yields 8.393 (+5.45%), while downstream benchmarks stay within ± 1 point of the base model. End-to-end QPU execution on `ibm_basquecountry` (156 qubits, Heron r2) was verified at increasing scale: a single block in ~ 4 s, one token (16 packed circuits) in 3 min 6 s, and the full 83-token sequence (1,328 circuits) in ≈ 4 h 24 min (Fig. 2c).

Systematic study: SmolLM2. We compress SmolLM2-135M [15] to 94.8M parameters with

Table 2: SmolLM2 (210-layer adapter) WikiText PPL vs. block size on the compressed backbone (PPL 35.31; teacher 24.10). nQ : qubits per block.

nQ	Block	Unconstr.	Unitary	Sign-constr.
2	4	34.23	34.73	34.68
4	16	33.25	33.91	34.01
6	64	31.37	32.47	32.88
8	256	28.83	30.25	31.25
input	384–896	27.07	27.97	29.94

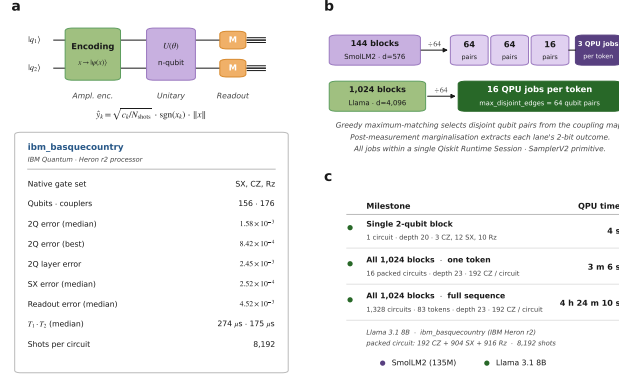


Figure 2: Realisation of the CUA on ibm_basquecountry (IBM Heron r2). (a) Two-qubit encode–unitary–measure pipeline and device parameters. (b) Circuit packing via greedy maximum-matching of disjoint two-qubit lanes. (c) QPU execution milestones for Llama 3.1 8B, from a single two-qubit block to the full 83-token autoregressive sequence.

CompactifAI [14] (WikiText PPL 24.10 $\rightarrow$ 35.31), then recover the loss with CUAs. Noiseless PPL improves *monotonically* with block dimension across all 210 layers with no saturation: input-dimension unitaries recover 48% of the compression gap (PPL 29.94), and sign-constrained (QPU-compatible) unitaries match unconstrained dense matrices at $\sim 50\%$ fewer parameters (Table 2); LAMBADA recovery reaches 83% (272.18 $\rightarrow$ 46.20). An ablation confirms the gain is specific to the *learned* structure: random adapters degrade PPL, while identity initialisation reproduces the baseline. Under per-gate depolarising noise the two-qubit regime (sub-1% total gate infidelity) preserves coherent generation, but at three qubits (an order of magnitude higher infidelity, 233 native gates) language structure collapses—defining the exact-synthesis feasibility frontier at 2–3 qubits for current hardware.

Entanglement structure. For the Llama v_proj-7 BDU (1,024 $SO(4)$ blocks, treated as a 12-qubit unitary), every block is individually entangling and the full BDU is *rank-saturating* across its natural block-index bipartition (operator-Schmidt rank 16/16). A depth-1 brickwork VQC at the same cut has rank 1 and needs depth ≥ 4 to match: the CUA is a strictly different ansatz class from a depth-matched VQC.

4 Discussion

The 4×4 circuits run here are classically simulable, so the QPU run is a hardware-feasibility demonstration rather than a claim of computational advantage—analogous in spirit to the NMR realisation of Shor’s algorithm [16]. The expressivity gain is nonetheless real and unsaturated (Table 2; a 4.74 PPL gap between two-qubit and full-dimension unitaries at $d = 576$) and is expected to widen at production scale; the route beyond two qubits is approximate compilation, motivated by the modest operator entropy of the trained unitaries, with the full advantage awaiting fault tolerance. CUAs thus give a concrete, hardware-efficient route to quantum-enhanced generative models across scales spanning two orders of magnitude.

References

- [1] T. B. Brown *et al.*, “Language models are few-shot learners,” *NeurIPS* **33**, 1877–1901 (2020).
- [2] Meta AI, “The Llama 3 herd of models,” arXiv:2407.21783 (2024).
- [3] J. Kaplan *et al.*, “Scaling laws for neural language models,” arXiv:2001.08361 (2020).
- [4] M. Cerezo *et al.*, “Variational quantum algorithms,” *Nat. Rev. Phys.* **3**, 625–644 (2021).
- [5] V. Havlíček *et al.*, “Supervised learning with quantum-enhanced feature spaces,” *Nature* **567**, 209–212 (2019).
- [6] B. Coecke, G. de Felice, K. Meichanetzidis, A. Toumi, “Foundations for near-term quantum natural language processing,” arXiv:2012.03755 (2020).
- [7] S. Yu *et al.*, “Quantum-enhanced large language model efficient fine tuning,” arXiv:2503.12790 (2025).
- [8] X. Zhao, H. Wu, L. Chen, “Training quantum self-attention on a 72-qubit quantum computer,” *IEEE Quantum Week* (2024).
- [9] B. Aizpurua, S. S. Jahromi, S. Singh, R. Orús, “Quantum large language models via tensor network disentanglers,” arXiv:2410.17397 (2024).
- [10] B. Aizpurua, S. Singh, R. Orús, “Classical neural networks on quantum devices via tensor network disentanglers,” arXiv:2509.06653 (2025).
- [11] A. Cayley, “Sur quelques propriétés des déterminants gauches,” *J. Reine Angew. Math.* **32**, 119–123 (1846).
- [12] M. Lezcano-Casado, D. Martínez-Rubio, “Cheap orthogonal constraints in neural networks,” *ICML*, 3794–3803 (2019).
- [13] G. Hinton, O. Vinyals, J. Dean, “Distilling the knowledge in a neural network,” arXiv:1503.02531 (2015).
- [14] A. Tomut *et al.*, “CompactifAI: Extreme compression of large language models using quantum-inspired tensor networks,” arXiv:2401.14109 (2024).
- [15] L. Ben Allal *et al.*, “SmolLM2: When smol goes big,” arXiv:2502.02737 (2025).
- [16] L. M. K. Vandersypen *et al.*, “Experimental realization of Shor’s quantum factoring algorithm using nuclear magnetic resonance,” *Nature* **414**, 883–887 (2001).