QKAN: Quantum Kolmogorov-Arnold Networks

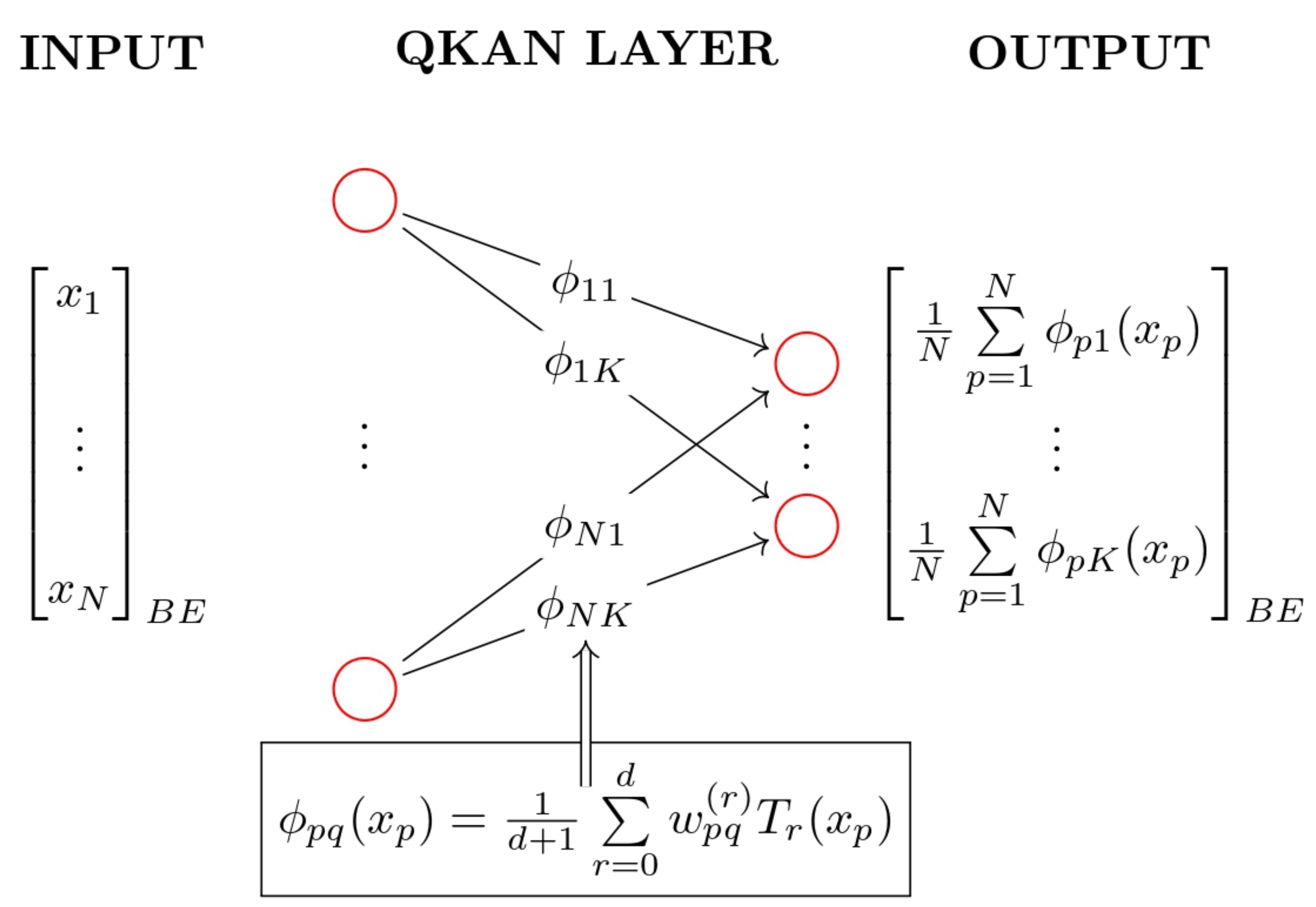
Petr Ivashkov,[1,2] Po-Wei Huang,[1] Kelvin Koor,[1] Lirandë Pira,[1] Patrick Rebentrost [1,3]

[1]Centre for Quantum Technologies, National University of Singapore, Singapore
[2]Department of Information Technology and Electrical Engineering, ETH Zurich, Zurich, Switzerland
[3]Department of Computer Science, National University of Singapore, Singapore

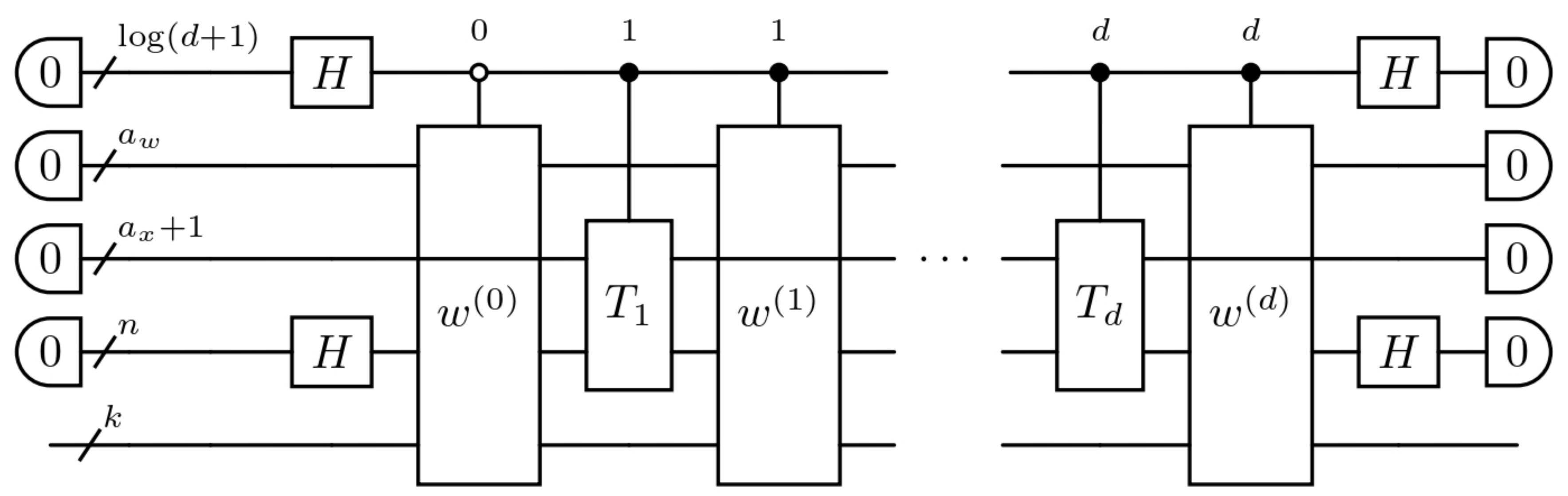
RESULT

We design a quantum version of Kolmogorov-Arnold networks [1] (KAN) called QKAN, based on block encodings and quantum linear algebra subroutines such as QSVT. QKAN takes as input a block-encoding of the input vector and outputs a block-encoding of the transformed vector. By integrating parameterized quantum circuits, QKAN can serve as both a trainable quantum machine learning model and a multivariate state preparation subroutine.

[1] Z. Liu et al., arXiv:2404.19756



CIRCUIT



The circuit corresnponds to a block-encoding of 2^k-dim output vector of a single-layer QKAN.

QKAN CONSTRUCTION

(1) Diagonal-block-encoding: Embed input vector diagonally into a unitary [2].

[2] A.G. Rattew, P. Rebentrost, arXiv:2309.09839

$$U_x = \left[\begin{array}{cccc|c} x_1 & 0 & \cdots & 0 & \\ 0 & x_2 & \cdots & 0 & * \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & \cdots & x_N & \\ \hline & * & & & * \end{array}\right] =: \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}_{\mathrm{BE}}$$

(2) CHEB: Apply Chebyshev polynomials $T_r(x)$ up to degree *d* via QSVT.

(3) MUL: Multiply each polynomial by the corresponding weight encoding.

(4) LCU: Combine weighted encodings with a Hadamard superposition.

(5) SUM: Sum activation outputs using Hadamards to yield final encoding.

(1) DBE **(2) CHEB** **(3) MUL** **(4) LCU** **(5) SUM**

$$\begin{bmatrix} x_1 \\ \vdots \\ x_1 \\ \vdots \\ x_N \\ \vdots \\ x_N \end{bmatrix}_{BE} \to \begin{bmatrix} T_r(x_1) \\ \vdots \\ T_r(x_1) \\ \vdots \\ T_r(x_N) \\ \vdots \\ T_r(x_N) \end{bmatrix}_{BE} \to \begin{bmatrix} w_{11}^{(r)}T_r(x_1) \\ \vdots \\ w_{1K}^{(r)}T_r(x_1) \\ \vdots \\ w_{N1}^{(r)}T_r(x_N) \\ \vdots \\ w_{NK}^{(r)}T_r(x_N) \end{bmatrix}_{BE} \to \begin{bmatrix} \phi_{11}(x_1) \\ \vdots \\ \phi_{1K}(x_1) \\ \vdots \\ \phi_{N1}(x_N) \\ \vdots \\ \phi_{NK}(x_N) \end{bmatrix}_{BE} \to \begin{bmatrix} \frac{1}{N}\sum_{p=1}^{N}\phi_{p1}(x_p) \\ \vdots \\ \frac{1}{N}\sum_{p=1}^{N}\phi_{pK}(x_p) \end{bmatrix}_{BE}$$

Complexity: An L-layer QKAN requires $\mathcal{O}\left(d^{2L}/2^L\right)$ applications of controlled block-encodings of input and weights.

APPLICATIONS

Regression (classical output)

Informal Theorem:

Given controlled block-encodings of input and weights

$$U_x : (1, a_x, \varepsilon_x), \quad U_{w^{(r)}} : (1, a_w, \varepsilon_w),$$

we can estimate each entry of a single QKAN layer

$$\Phi(\boldsymbol{x})_q = \tfrac{1}{N}\textstyle\sum_{p=1}^{N}\phi_{pq}(x_p)$$

to additive $(4d\sqrt{\varepsilon_x} + \varepsilon_w + \delta)$ - precision using* $\mathcal{O}\left(d^2/\delta\right)$ calls to *controlled-*U_x and *controlled-*$U_{w^{(r)}}$.

State-preparation (quantum output)

Informal Theorem:

Given controlled block-encodings of input and weights

$$U_x : (1, a_x, \varepsilon_x), \quad U_{w^{(r)}} : (1, a_w, \varepsilon_w),$$

we can prepare a quantum state $|\psi\rangle$ with amplitudes corresponding to a *K*-dim QKAN layer

$$\left\| |\psi\rangle - \frac{1}{N}\sum_{q=1}^{K}\left(\frac{1}{N}\sum_{p=1}^{N}\phi_{pq}(x_p)\right)|q\rangle_k \right\|_2 \le \varepsilon,$$

using* $\mathcal{O}(\sqrt{K}d^2/\mathcal{N})$ calls to *controlled-*U_x and *controlled-*$U_{w^{(r)}}$.

* *d* is the maximal degree of Chebyshev polynomials.

FAQ

- **What is QKAN?** A quantum neural-network architecture that emulates the classical KAN.
- **What is the core idea?** Interpret the eigenvalues of a block-encoded matrix as neural network nodes, then leverage QSVT to implement nonlinear activation functions.
- **What is the main advantage over classical KANs?** The runtime depends implicitly on the input dimension *N* through the cost of the block-encoding. A *polylog(N)* block-encoding potentially delivers an exponential speedup.
- **What is the main disadvantage?** The runtime grows exponentially with the number *L* of network layers.

ETHzürich

Centre for Quantum Technologies

NUS National University of Singapore