Kundu et al. *EPJ Quantum Technology* (2024) 11:76
https://doi.org/10.1140/epjqt/s40507-024-00289-z

EPJ Quantum Technology
a SpringerOpen Journal

RESEARCH Open Access

KANQAS: Kolmogorov-Arnold Network for Quantum Architecture Search

Akash Kundu[1,2,3,4*], Aritra Sarkar[4,5] and Abhishek Sadhu[6,7]

*Correspondence:
akash.kundu@helsinki.fi
[1]QTF Centre of Excellence, Department of Physics, University of Helsinki, Helsinki, Finland
[2]Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Gliwice, Poland
Full list of author information is available at the end of the article

Abstract
Quantum architecture Search (QAS) is a promising direction for optimization and automated design of quantum circuits towards quantum advantage. Recent techniques in QAS emphasize Multi-Layer Perceptron (MLP)-based deep Q-networks. However, their interpretability remains challenging due to the large number of learnable parameters and the complexities involved in selecting appropriate activation functions. In this work, to overcome these challenges, we utilize the Kolmogorov-Arnold Network (KAN) in the QAS algorithm, analyzing their efficiency in the task of quantum state preparation and quantum chemistry. In quantum state preparation, our results show that in a noiseless scenario, the probability of success is 2× to 5× higher than MLPs. In noisy environments, KAN outperforms MLPs in fidelity when approximating these states, showcasing its robustness against noise. In tackling quantum chemistry problems, we enhance the recently proposed QAS algorithm by integrating curriculum reinforcement learning with a KAN structure. This facilitates a more efficient design of parameterized quantum circuits by reducing the number of required 2-qubit gates and circuit depth. Further investigation reveals that KAN requires a significantly smaller number of learnable parameters compared to MLPs; however, the average time of executing each episode for KAN is higher.

Keywords: Kolmogorov-Arnold network; Quantum architecture search; Quantum state reconstruction; Quantum chemistry

1 Introduction

Recent research has advanced quantum computing concepts and software development, yet, significant challenges remain before real-world applications are feasible. Automating quantum algorithm design via machine learning and optimization algorithms offers promising solutions to enhance quantum hardware and programming capabilities for complex problems. Strongly inspired by neural architecture search [1], Quantum Architecture Search (QAS) [2, 3] is introduced.

QAS encompasses techniques to automate the optimization of quantum circuits and it typically consists of two parts. In the first part, a template of the circuits is built where the circuit can have parameterized quantum gates, e.g., rotation angles. Then, these parameters are determined via the variational principle using a classical optimizer in a feedback loop. Algorithms constructed via this technique are called Variational Quantum Algorithms (VQA) [4, 5]. The parameterized circuit design in VQAs is critical, as it directly influences the expressivity and efficiency of the quantum solution. Hence to automate the search for an effective circuit, in the next part, a search algorithm is employed to decide the quantum gates and the corresponding parameters. Recently, QAS has been utilized to automate the search for optimal parameterized circuits for VQAs. QAS also finds application in determining non-parameterized circuits as an approach for quantum program synthesis [6] and multi-qubit maximally entangled state preparation [7].

One of the most prominent methods to tackle QAS problems is to use Reinforcement Learning (RL) (RLQAS) [7–12] as the search algorithm. In this approach, quantum circuits are defined as a sequence of actions generated by a trainable policy. The cost function's value, optimized independently by a classical optimizer, serves as a signal for the reward function. This reward function guides the policy updates to maximize expected returns and select optimal actions for subsequent steps. Maximization of the expected return is achieved by training Neural Networks (NNs). Recently, NNs have shown potential in quantum information tasks [13, 14]. RLQAS with deep NNs can overcome the trainability issues of VQAs [15, 16] and has demonstrated promising outcomes in NISQ hardware [10, 11].

In a recent work [17], researchers have introduced Kolmogorov-Arnold Networks (KANs) as a novel neural network architecture typically poised to replace traditional Multi-Layer Perceptrons (MLPs). Contrary to the universal approximation theorem-based MLPs, KAN utilizes the Kolmogorov-Arnold representation theorem to approximate complicated functions. KAN has linear weights replaced by spline-based univariate functions along the network edges and is structured as learnable activation functions. Such a design enhances the accuracy and interpretability of the networks, enabling them to achieve comparable or superior results with smaller network sizes across a range of tasks, including data fitting and solving partial differential equations. Recently, different variants of KAN have been introduced [18–23]. KAN has applications in time series analysis [24–26], satellite image classification [27], and improving the robustness of neural network architectures [28]. To understand the full potential and limitations of KANs there is a need for further investigation towards robustness and compatibility with other deep learning architectures.

In this article, we evaluate the practicality of KANs in quantum circuit construction, analyzing their efficiency in terms of the probability of success, frequency of optimal solutions and their dependencies on various degrees of freedom of the network.

To the best of our knowledge, the application of Kolmogorov-Arnold Networks in Quantum Architecture Search is still lacking in standard literature. *We propose the application of KAN in QAS by replacing the MLP of Double Deep Q-Network (DDQN) in RLQAS with KAN to generate the desired quantum state, introducing KANQAS.*[1] As shown in Fig. 1, the proposed framework of KANQAS features an RL-agent, which contains the KAN, interacting with a quantum simulator. The agent sequentially generates output actions, which are candidates for quantum gates on the circuit. The fidelity of the state from the constructed circuit is compared to the quantum state fidelity of the circuit and is evaluated to determine how far it is from the desired goal. The reward, based on fidelity, is sent back to the RL-agent. This process is repeated iteratively to train the RL-agent. We show that in a noiseless scenario when constructing Bell and Greenberger–Horne–Zeilinger (GHZ)

[1] Link to KANQAS code repository.

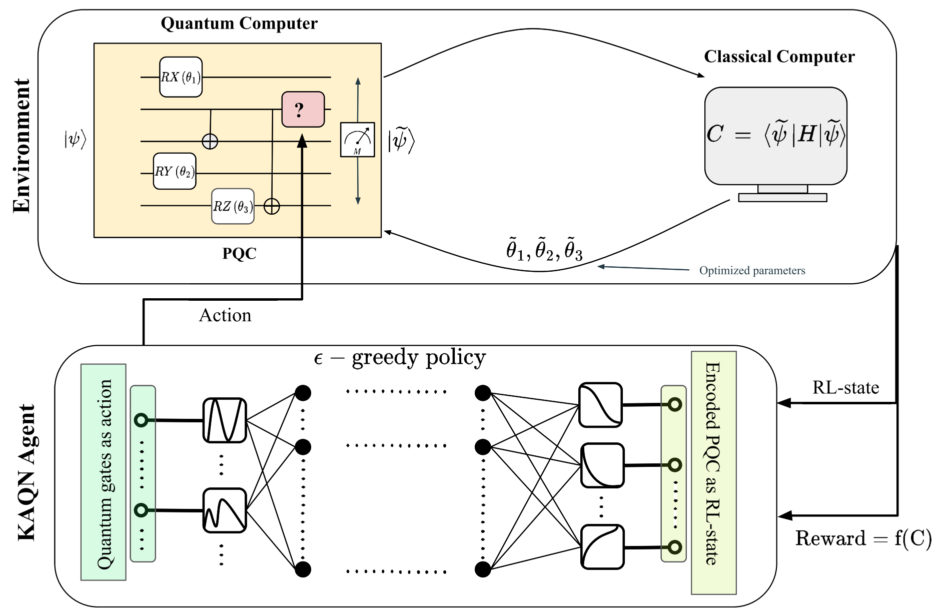


Figure 1 The schematic for KANQAS illustrates how the Kolmogorov-Arnold Network (KAN) replaces the traditional multi-layer perceptron in the Reinforcement Learning (RL) agent. In this setup, the environment, which incorporates a quantum algorithm, interacts with the RL-agent powered by a KAN-driven double deep-Q network, referred to as KAQN. Following an ϵ-greedy policy, the agent selects its next action based on the reward function and the RL-state received from the environment. For the details of the reward function construction and quantum circuit encoding into an RL-state check the Sect. 4 and Appendix A.1 respectively

states, the probability of success and the number of optimal quantum circuit configurations to generate the states are significantly higher than MLPs. In a noisy scenario, we show that KAN can achieve a better fidelity in approximating GHZ state than MLPs, where the performance of the MLP significantly depends on the depth of the network and choice of activation function. We also address quantum chemistry problems, where we find the ground state of 4-qubit $\mathtt{H}_2$ and $\mathtt{LiH}$ molecules using recently proposed Curriculum Reinforcement Learning for Quantum Architecture Search (CRLQAS) [11]. In our version of CRLQAS, we replace the MLP structure with KAN in the curriculum reinforcement learning subroutine. Our results show that KAN demonstrates the ability to provide a more compact parameterized quantum circuit, in terms of less number of 2 qubit gates and depth for finding the ground state of the chemical Hamiltonian using VQE, with a significantly reduced number of learnable parameters.

The structure of the paper is as follows. Section 2 discussed key related developments in QAS. We present the problem statement in Sect. 3. Section 4 provides the methods used in this work. Specifically, we introduce the Kolmogorov-Arnold Q Network in Sect. 4.1 and discuss the RL-state, action and reward function in Sect. 4.2. We present our results in Sect. 5. Specifically, we discuss the results for state preparation using noiseless simulations in Sect. 5.1.1, noisy simulations in Sect. 5.1.2 and for quantum chemistry simulation in Sect. 5.2. The resource requirements for the simulations in estimated in Sect. 6. We provide concluding remarks and discuss open problems in Sect. 7.

2 Related works

In the past few years, several key developments in the field of Quantum Architecture Search (QAS) have emerged that aim to enhance the efficiency and performance of these algorithms. The majority of these research works, in a nutshell, tackle the QAS problem by either developing sophisticated optimization strategies or utilizing well-established methods from classical machine learning such as Reinforcement Learning (RL).

In [29], the authors introduce a framework that utilizes neural networks to predict the performance of quantum circuit architectures, significantly enhancing the efficiency of the architecture search process. In [2], the authors utilize Monte Carlo sampling to search for Parameterized Quantum Circuits (PQCs) to solve combinatorial optimization problems. Following this line of research, Ref. [30] utilizes the Gumbel-Softmax sampling technique to sample quantum circuits and benchmark the QAS on quantum chemistry tasks. In [10], a QAS method based on supernet and weight sharing strategy was introduced to better estimate energy for quantum chemistry tasks. In [31], by employing a "SuperCircuit" encapsulating various PQCs, QuantumNAS facilitates an efficient search for optimal circuit architectures and qubit mappings to solve quantum chemistry problems. In [32], the authors incorporate the information corresponding to the gradient of the PQCs, which reduces the computational burden associated with evaluating candidate architectures in QAS, allowing for more effective exploration of the design space in quantum computing tasks. To bypass the training subroutine of different quantum architectures in [33] the authors propose a proxy-based training-free QAS algorithm. Where among the two proxies, the first one eliminates the unpromising PQCs and the second proxy, based on the expressibility of the PQCs to assess their performance. Moreover, in [34], a framework is proposed that automates the design of quantum circuit structures for interconnected quantum processing units, incorporating innovative methods for nonlocal gate implementation and qubit assignment to enhance computational efficiency.

Reinforcement Learning (RL) based QAS namely, *RLQAS* has also been considered to automate the search for optimal PQCs. Typically, RL approaches employ a carefully designed reward function to train the agent to choose suitable gates. In [8] the authors employed Double Deep Q-Network (DDQN) to estimate the ground state of chemical Hamiltonians. Following this line of approach in [11], the authors tackle QAS problems under realistic quantum hardware. This is achieved by introducing a tensor-based encoding for PQCs, a gate-set pruning approach, and a sophisticated second momentum-based classical optimization method, as well as by minimizing environment-agent interaction. In [9], by utilizing a novel encoding method for the PQCs, a dense reward function, and an ϵ-greedy policy, the authors tackle the quantum state diagonalization problem. Additionally, in [35], the authors show that by utilizing RL, it is possible to solve the hard instances of combinatorial optimization problems where state-of-the-art algorithms perform suboptimally. Finally, in [12], the authors leverage insights from quantum information theory, which helps the RL-agent to prioritize certain architectural features that are likely to provide better performance in QAS.

For a more brief overview of QAS approaches we encourage the authors to check the refs. [36–38].

3 Problem statement

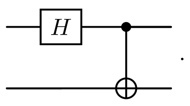
In the previous section, we observed that significant research has focused on leveraging classical neural network architectures to improve the performance of RLQAS methods. While Multi-Layer Perceptron (MLP) structures in RL offer advantages, the number of trainable parameters grows rapidly with problem size, and their performance is highly sensitive to the choice of activation function. This work addresses the question: *Can Kolmogorov-Arnold Network (KAN) [17] replace the MLP structure in RLQAS to achieve comparable or better performance while mitigating the issues of parameter growth and activation function selection?* We demonstrate that the answer is affirmative by tackling two crucial problems in quantum computation which are described in the following sections.

3.1 Quantum state preparation

To evaluate the efficiency of RL-assisted KANQAS in quantum circuit construction we check if, in a noiseless and noisy scenario, multi-qubit entanglement can be generated as expected. To this end, we target the generation of two kinds of quantum states: Bell and Greenberger–Horne–Zeilinger (GHZ) states. A Bell state is a maximal 2-qubit entangled state and is given by

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right). \tag{1}$$

The optimal circuit to produce a Bell state is given by

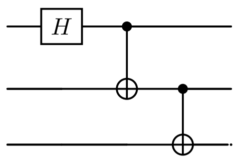


(2)

Meanwhile, a GHZ state is a 3-qubit maximally entangled state given by

$$|GHZ\rangle = \frac{1}{\sqrt{2}}\left(|000\rangle + |111\rangle\right), \tag{3}$$

and the optimal circuit produces a GHZ state given by



(4)

As a measure of the efficiency of the network, we define the probability of success and the probability of optimal success. The probability of success is given by

$$\frac{\text{Number of times the network finds an admissible circuit}}{\text{Total number of admissible circuits}}, \tag{5}$$

As the problem statement clarifies, any circuit that generates a Bell state and a GHZ state is considered an admissible circuit. Meanwhile, Fig. 2 and 4 are optimal admissible circuits to generate 2- and 3-qubit maximally entangled state.

3.2 Quantum chemistry

Quantum chemistry [39] utilizes quantum mechanics to understand the behavior of atoms and molecules. Its main goal is to understand the electronic structure of chemical systems and dynamics, enabling predictions about their chemical and physical properties. This involves examining the electronic ground and excited states, reaction pathways, and transition states, crucial for explaining reactivity and interactions within chemical systems.

The Variational Quantum Eigensolver (VQE) is a quantum-classical algorithm that is designed to find the ground state energy of quantum systems, making it particularly useful for studying molecular systems where traditional computational methods may struggle due to the complexity and the exponentially growing size of the systems.

In VQE, the objective is to find the ground state energy of a chemical Hamiltonian H by minimizing the energy

$$C(\vec{\theta}) = \min_{\vec{\theta}} \left(\langle \psi(\vec{\theta}) | H | \psi(\vec{\theta}) \rangle \right). \tag{6}$$

The trial state $|\psi(\vec{\theta})\rangle$ is prepared by applying a Parameterized Quantum Circuit (PQC), $U(\vec{\theta})$, to the initial state $|\psi_{\text{initial}}\rangle$, where $\vec{\theta}$ specify the rotation angles of the local unitary operators in the circuit.

The structure of the PQC is one of the crucial factors affecting the performance of the VQE. Among the recently proposed RLQAS methods for automating the search for PQC in VQE, Curriculum Reinforcement Learning (CRL) for QAS, i.e., CRLQAS [11] demonstrates promising outcomes to find the ground state of various chemical Hamiltonian in the noiseless scenario and under hardware error with short PQCs.

Hence, to evaluate the efficiency of KAN in PQC construction for quantum chemistry problems we replace the MLP structure utilized in CRLQAS with KAN. Utilizing the CRL-assisted KANQAS we find the ground state of 4-qubit $\mathtt{H_2}$ and $\mathtt{LiH}$ molecules. The exact molecular structures of these molecules are provided in Table 1.

4 Methods

4.1 Kolmogorov-Arnold Q Networks (KAQN)

In a recent work [17], Kolmogorov-Arnold Networks (KAN) was proposed as a promising alternative to the Multi-Layer Perceptrons (MLP). KAN is based on the Kolmogorov-Arnold representation theorem [40] instead of the Universal Approximation Theorem [41] used in MLP. The Kolmogorov-Arnold representation theorem states that a real-valued, smooth and continuous multivariate function $g(\mathbf{t}) : [0,1]^n \to \mathbb{R}$ can be represented by a superposition of univariate functions [40]

$$g(\mathbf{t}) = \sum_{j=1}^{2n+1} \Psi_j \left(\sum_{k=1}^{n} \psi_{jk} \right), \tag{7}$$

Table 1 List of molecules included in our simulations, with configuration coordinates provided in angstroms

Molecule	Basis	Fermion to qubit mapping	Geometry	Number of qubits
$\mathtt{H_2}$	sto3g	Jordan-Wigner	$\mathtt{H}$ (0, 0, −0.35); $\mathtt{H}$ (0, 0, 0.35)	4
$\mathtt{LiH}$	sto3g	Parity	$\mathtt{Li}$ (0, 0, 0); $\mathtt{H}$ (0, 0, 3.4)	4

where $\Psi_j : \mathbb{R} \to \mathbb{R}$ and $\psi_{jk} : [0,1] \to \mathbb{R}$. In other words, any multivariate continuous function on a bounded domain can be expressed as a composition of continuous functions of one variable. This reduces the task of learning complex multi-variable functions to learning a polynomial number of single-variable functions. It was noted by the authors of [17] that the representation of the function in Eq. (7) has two layers of nonlinearity with $2n+1$ terms in the middle layer, and we need to find functions Ψ_i and ψ_{ij} to approximate $g(\mathbf{t})$. The function ψ_{ij} may be approximated using B-splines [42].

A spline is a piecewise smooth polynomial function defined by a set of control points. It is defined by the order l of the polynomial used to interpolate the curve between the control points. We denote the number of segments between adjacent control points as G. The data points are connected by the segments to form a smooth curve having $G+1$ grid points. It is observed that Eq. (7) can be represented as a two-layer network having activation functions at the edges and nodes performing the summation operation. However, such a network is too restrictive to approximate any arbitrary function. A way to overcome this was proposed in [17], where the authors propose a general architecture with wider and deeper KANs.

The authors of [17] define a KAN layer by a matrix $\boldsymbol{\Psi}$ of trainable univariate spline functions $\{\psi_{jk}(.)\}$ with $j = 1, \ldots, n_i$ and $k = 1, \ldots, n_o$, where n_i and n_o denotes the number of inputs and outputs respectively. The Kolmogorov-Arnold representation theorem can be expressed as a two-layer KAN. The inner functions constitute a KAN layer with $n_i = n$, $n_o = 2n+1$ while the outer function is another KAN with $n_i = 2n+1$, $n_o = n$. We define the shape of a KAN by $[n_1, \ldots, n_k]$ with k denoting the number of layers of the KAN. The Kolmogorov-Arnold representation theorem can be expressed as a KAN of shape $[n, 2n+1, 1]$. A deeper KAN can be expressed by the composition k layers:

$$\mathbf{z} = \mathrm{KAN}(\mathbf{t}) = (\boldsymbol{\Psi}_k \circ \boldsymbol{\Psi}_{k-1} \circ \cdots \circ \boldsymbol{\Psi}_1)\mathbf{t}. \tag{8}$$

Since all functions are differentiable, KAN can be trained using backpropagation [43]. For the sake of simplicity, we describe a 2-depth KAN as $[n_i, n, n_o]$, where the input layer is not included in the depth count. The output and input layers will be comprised of n_o, and n_i nodes corresponding to the total amount of time steps while n describes the number of the hidden layers.

KAN can learn features and compositional structure due to their outer structure resembling MLPs and optimize the learned features by approximating the univariate functions with high accuracy due to their internal spline structure. It should be noted that increasing the number of layers of the dimension of the grid increases the number of parameters and, hence, the complexity of the network.

Motivated by the developments in [44], we replace the Multi-Layer Perceptron (MLP) component of Deep Q-Networks (DQN) with the KAN. Furthermore, we employ the Double Deep Q-Network (DDQN) update rule to enhance stability and learning efficiency. DDQN [45] is a Q-learning algorithm based on standard DQN [46], which features two neural networks to increase the stability of the prediction of Q-values for each state and action pair. For more details, see Appendix A.2.

4.2 RL-state, action space and reward function

The RL environment in KANQAS is encoded using the tensor-based one-hot encoding method described in [11]. However, the tensor's dimensions vary depending on the size of the action space. The RL-state encoding translates a quantum circuit into a tensor of

$$\text{size} = D_{\text{max}} \times N \times (N + N_{1\text{q}}), \tag{9}$$

where D_{max} is a hyperparameter and is defined as the maximum allowed gates per episode i.e. the length of an episode, N is the number of qubits and $N_{1\text{q}}$ defines the number of 1-qubit gates. The first $N \times N$ encodes the position of the 2-qubit gate and the remaining $N \times N_{1\text{q}}$ encodes the position of the 1-qubit gate. For a brief discussion on the quantum circuit encoding see Appendix A.1.

Meanwhile, the definition of the reward function varies depending on the problem under consideration. These aspects are further elaborated in the following sections.

Quantum state preparation In this problem, we initialize with an empty quantum circuit (i.e. without any gates). Depending on a fidelity-based reward function of the form

$$R = \begin{cases} \mathcal{R}, & \text{if } F(s_t) \geq 0.98 \\ F(s_t), & \text{otherwise} \end{cases} \tag{10}$$

and by following an ϵ-greedy policy the RL-agent sets the probability of selecting a random action. Where $F(s_t)$ is the fidelity of a state at step t generated by KANQAS and $\mathcal{R} \gg F(s_t)$, is a hyperparameter.

The random action is chosen from a predefined action space ($\mathbb{A}$) which contains non-parametrized 1- and 2-qubit gates [7]

$$\mathbb{A} = \{\texttt{CX}, \texttt{X}, \texttt{Y}, \texttt{Z}, \texttt{H}, \texttt{T}\}. \tag{11}$$

Depending on the action the RL-state, which is encoded into a tensor of dimension $D_{\text{max}} \times N \times (N+5)$, is modified in the next step. For further elaboration, check the detailed discussion of the encoding of PQC provided in Appendix A.1.

Quantum chemistry Following the same principle as discussed in the previous section, in this problem, we initialize with an empty quantum circuit (i.e. without any gates). However, for a fair comparison with MLP bases CRLQAS, we utilize a reward function of the form

$$R = \begin{cases} 5 & \text{if } C_t < \xi, \\ -5 & \text{if } t \geq D_{\text{max}} \text{ and } C_t \geq \xi, \\ \max\left(\frac{C_{t-1}-C_t}{C_{t-1}-C_{\text{min}}}, -1\right) & \text{otherwise} \end{cases} \tag{12}$$

where C_t is the VQE cost function (see Eq. (6)) at RL step t and ξ is a hyperparameter, but for VQE it is the chemical accuracy 0.0016 Hartree. D_{max} is a hyperparameter that defines the maximum number of actions per episode, i.e. length of an RL-episode. The random action is chosen from a predefined action space ($\mathbb{A}$) which contains parametrized 1- and non-parameterized 2-qubit gates [8, 9, 11]

$$\mathbb{A} = \{\texttt{CX}, \texttt{RX}, \texttt{RY}, \texttt{RZ}\}. \tag{13}$$

Depending on the action, the RL-state, which is encoded into a tensor of dimension $D_{\max} \times N \times (N + 3)$, is modified in the next step. An elaborated discussion of the encoding is provided in Appendix A.1.

Throughout the paper, the configurations of KANs and MLPs are written using the short notation: KAN_{l+1,n_h} for a KAN and MLP_{l+1,n_h} for an MLP where l number of hidden layers in the neural network, and n_h is the number of neurons per hidden layer. For example, a KAN with the structure $[n_i, n_h^{(1)}, n_h^{(2)}, \ldots, n_h^{(l)}, n_o]$ is written as KAN_{l+1,n_h}, where the depth of the network is $l + 1$ and each hidden layer has n_h neurons. n_i is calculated using the Eq. (9), for an example while tackling the 2-qubit ($N = 2$) state reconstruction problem, we consider an action space consisting of five 1-qubit gates i.e. $N_{1\text{q}} = 5$ and one 2-qubit gate with the maximum achievable depth $D_{\max} = 6$. Hence, following Eq. (9), we get $n_i = 84$. To solve the problem, we consider a KAN of depth 2, and the number of neurons per hidden layer is $n_h = 3$, i.e. $[84, 3, 12]$. The size of the output is 12 because, for 2-qubit, the $\texttt{CX}$ gate has two variants $\texttt{CX}_{12}$ and $\texttt{CX}_{21}$ where the target in the later is on the first and the former is in the second qubit. Meanwhile, as the 1-qubit gates can be added on both qubits so, for five 1-qubit gates, we have 10 actions, hence a total of 12 actions.

5 Results

In this section, we benchmark the performance of KANQAS for quantum state preparation and quantum chemistry tasks and show that, in many instances, a KAN configuration outperforms an MLP with a lesser number of trainable parameters. For the quantum state preparation, we consider the task of finding a 2- and 3-qubit maximally entangled state in the noiseless and noisy scenario and for quantum chemistry, we use the KAN and MLP structures to find the ground state of 4-qubit $\texttt{H}_2$ and $\texttt{LiH}$ molecules.

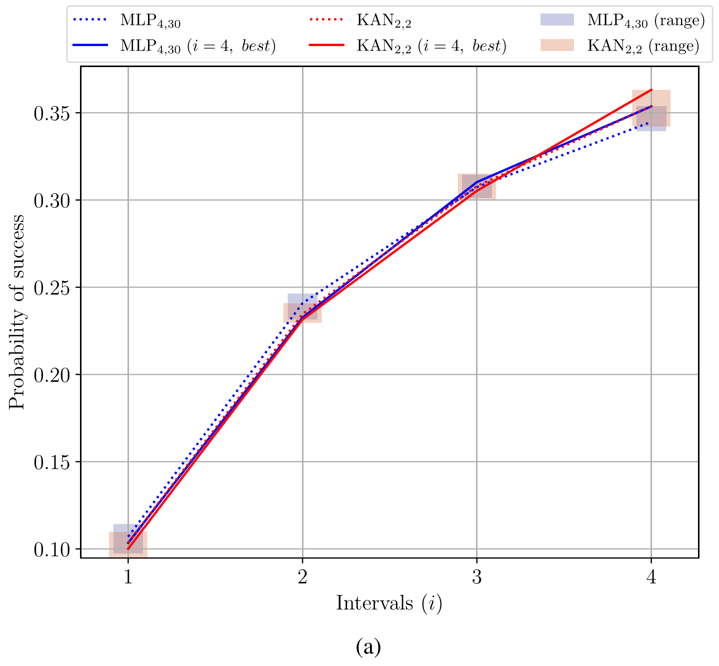
5.1 Quantum state preparation

5.1.1 Noiseless simulation

In the noiseless case, we consider the structure of the KAN and the MLP as presented in Table 5. One of the prominent advantages of KAN is that its activation function is learnable; meanwhile, in the case of MLP, the activation function is a hyperparameter. In the upcoming sections, we will see that the performance of the MLP is heavily dependent on the choice of the activation function.

For the construction of Bell state, we run a total of 10,000 episodes where within each episode, we allow the agent to create a quantum circuit of maximum depth 6, i.e. $D_{\max} = 6$. For a better representation of the results, we divide the total number of episodes into 4 intervals, where each interval contains 2500 episodes. Figure 2(a) investigates the probability of success in each interval, which is calculated using (5). Meanwhile, in Fig. 2(b), we see the variation in the number of optimal admissible circuits in each interval. We call a quantum circuit an admissible circuit when the state generated by it obtains a positive reward based on the reward function in Eq. (10).

In each interval, the probability is averaged over 20 random seeds, where each seed corresponds to the random initialization of the network. We observe that the probability of success with an MLP and KAN is comparable in the first 3 intervals, but in the 4th interval, i.e., in the episode range of 7500 – 10,000, the probability of success achieved by KAN is higher. With MLP, the best probability of success achievable in the 4th interval is 35.36% whereas with KAN, we can achieve a success probability 36.31%, whereas the number of



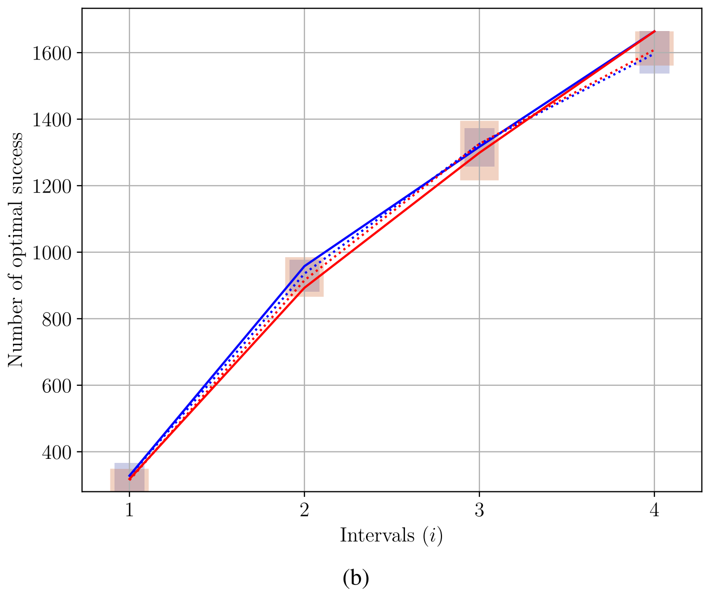


Figure 2 In (a) the probability of successful circuits and in (b) the probability of optimal successful circuits in finding a 2-qubit maximally entangled state is slightly higher with KAN than MLP. A total of 10,000 episodes are divided into 4 separate intervals where each interval contains 2500 episodes. In (a), each point in an interval corresponds to the probability of occurrence of a successful episode (see Eq. (5)). Similarly, (b) corresponds to the number of occurrences of an optimal successful episode. The results are averaged over 20 random seeds (i.e. initialization) of the networks

optimal admissible ansatz achievable by both networks is the same. This indicates that KANs have the potential to generate more diverse solutions to the same problem than MLPs.

Meanwhile, the difference in performance between KAN and MLP becomes significant in the task of constructing the GHZ state. Here we consider a depth 2 (containing a single hidden layer) MLP with 3 neurons ($\text{MLP}_{2,3}$) and depth 4 (containing 3 hidden layers) MLP with 30 neurons ($\text{MLP}_{4,30}$) and compare its performance with a depth 2 KAN containing 3 neurons ($\text{KAN}_{2,3}$). We run a total of 8000 episodes where in each episode, the RL-agent is allowed to make a quantum circuit of maximum depth $D_{\text{max}} = 12$. Just like the previous 2-qubit experiment, we divide the total number of episodes into 4 intervals where each interval contains 2000 episodes.

In Fig. 3(a) we observe that the average probability of success with $\text{KAN}_{2,3}$ is higher than both $\text{MLP}_{2,3}$ and $\text{MLP}_{4,30}$ over 15 random initialization of the networks. This difference becomes truly significant when we consider the best performance at the 4th interval. *With $KAN_{2,3}$ we can achieve a probability of success in the final interval 48.41% whereas with $MLP_{2,3}$ and $MLP_{4,30}$ we get 38.23% and 38.61% respectively, indicating a $10.18 - 9.8\%$ performance enhancement with KAN.*

In Fig. 3(b), we observe that on average (over 15 random initialization of the networks), one can achieve a higher number of optimal admissible ansatz using $\text{KAN}_{2,3}$ than the MLPs. Meanwhile, when we consider the best performance in the 4th interval *the number of optimal admissible ansatz achievable by KAN is $1.84\times$ to $5.16\times$ higher than $MLP_{2,3}$ and $MLP_{4,30}$ respectively*. Noticing the significant improvement of RLQAS with KAN in the noiseless scenario, in the following section, we focus on a more realistic setting where quantum gates are subject to noise.

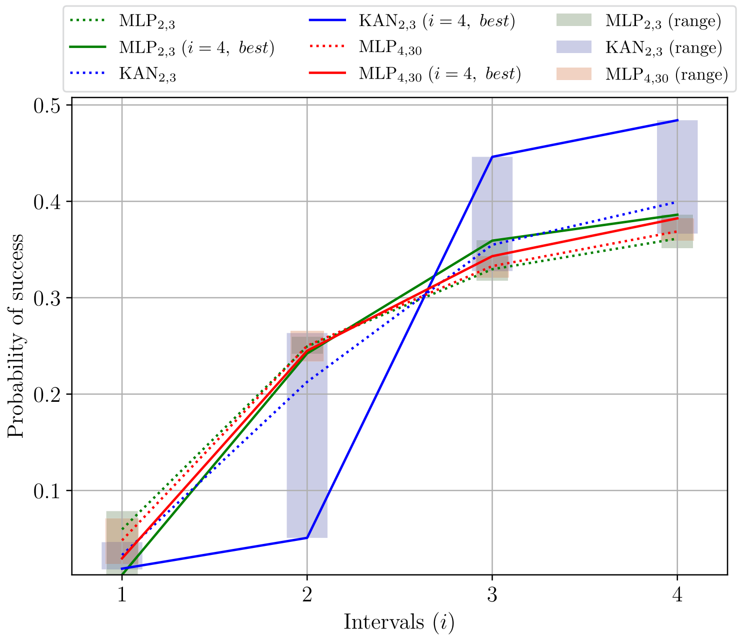
5.1.2 Noisy simulation

For the noisy simulation of Bell state preparation, we consider two forms of gate errors. The gate error refers to the imperfection in any quantum operation we perform. For the 1-qubit gate error, we consider random X noise where with probability p_{x} an X gate is applied to the circuit and with $1 - p_{\text{x}}$ it applies an $\mathbb{I}$. Meanwhile, for 2-qubit gate error, we apply depolarizing noise which replaces the state of any qubit with a random state of probability p_{dep}. Under these noisy scenarios, we utilize different configurations of MLP and KAN (summarized in Table 2) to reconstruct the Bell state.

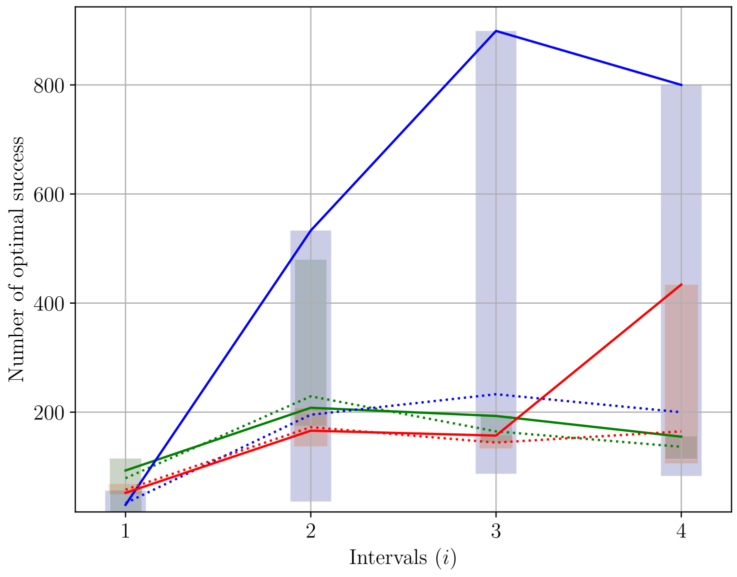
At the first stage, we consider noise levels $p_{\text{x}} = 0.1$ and $p_{\text{dep}} = 0.01$, under these circumstances the $\text{MLP}_{4,30}$ (i.e. an MLP of depth 4 and 30 neurons) can achieve a fidelity 99.25% whereas the same fidelity can be achieved with KAN with just depth 2 and 2 neurons. Now to elevate the hardness of the problem, in the second experiment, we consider the following noise levels: $p_{\text{x}} = 0.3$ and $p_{\text{dep}} = 0.2$. With KAN configuration presented in Table 2 we can achieve a fidelity of 73.28% which $\text{MLP}_{4,30}$ with ReLU and LeakyReLU activation functions cannot achieve. Even when the number of neurons is increased tenfold compared to KAN, $\text{MLP}_{4,100}$ (i.e., MLP with depth 4 and 100 neurons) still fails to surpass KAN's fidelity using ELU and ReLU activation functions. However, with LeakyReLU activation, it achieves a higher fidelity of 85%.

This leads to the conclusion that to achieve competitive/better performance with an MLP, it is necessary to fine-tune not just the network's depth and width but also the activation function. However, with KAN, this process becomes much more straightforward, as the activation functions are learnable.

One can argue that KAN has two additional parameters: splines (k) and grid (G), and tuning these hyperparameters can heavily affect its performance. In Fig. 4, for constructing GHZ state, we show that *to achieve better performance in terms of the depth and number of gates variation in the number of splines (i.e. k) is more effective and stable for the network*



(a)



(b)

Figure 3 In (a) the probability of successful circuits and in (b) the total number of optimal successful circuits in finding a 3-qubit maximally entangled state are noticeably higher with KAN than MLP. A total of 8000 episodes are divided into 4 separate intervals, where each interval contains 2000 episodes. In (a), each point in an interval corresponds to the probability of occurrence of a successful episode (see Eq. (5)). The results are averaged over 15 random seeds (i.e. initialization) of the networks. The *range* defines the best performance of each interval for both networks. The *range* defines the region between the best and the worst performance in each interval for both networks

than variation in G. Whence, in ref. [26], the authors show that achieving a better minimization of the loss function grid size within the splines of KANs has a notable impact.

Table 2 KAN outperforms most of the configurations of MLPs except when $MLP_{4,100}$ is operated with LeakyReLU activation function for noisy Bell state preparation with $p_x = 0.3$ and $p_{dep} = 0.2$. This helps us conclude that to achieve competitive/better performance with an MLP, it is necessary to fine-tune not just the network's depth and width but also the activation function In configuration [84, 10, 10, 12] the KAN takes 84 inputs that encode the quantum circuit, calculated using Eq. (9) for $D_{max} = 6$, $N = 2$ and $N_{1q} = 5$ and outputs 12 actions after passing through 2 hidden layers containing 10 neurons

Network	Configuration	Spline and grid	Activation func.	Fidelity
$KAN_{3,10}$	[84, 10, 10, 12]	B-spline, $k = 4$, $G = 5$	learnable	0.7328
$MLP_{4,30}$	[84, 30, 30, 30, 12]	NA	ReLU	0.5005
		NA	LeakyReLU	0.7300
$MLP_{4,100}$	[84, 100, 100, 100, 12]	NA	ELU	0.6830
			ReLU	0.7300
			LeakyReLU	**0.8500**

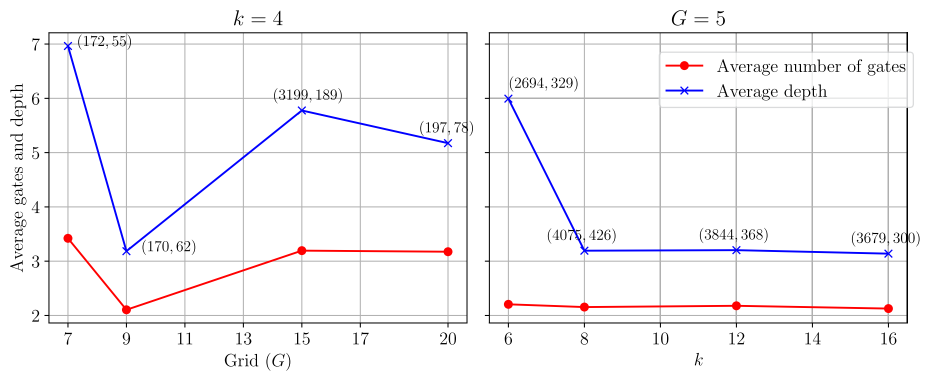


Figure 4 The number of splines (k) has more impact in improving and stabilizing the performance of the KAN than the grid size (G). Here we measure the improvement of the KAN while constructing the GHZ state by calculating the average number of gates and depth in each setting. Each point is marked as (a, b) where a is the total number of successful episodes and b total optimal successful episodes

Meanwhile, loss function minimization is crucial for Variational Quantum Algorithms (VQAs), so it is much needed to fine-tune the grid size while optimizing parameterized gates within Parameterized Quantum Circuits (PQCs) of many VQAs. Following these developments in the upcoming section, we tackle the construction of a PQC that can construct the ground state of chemical Hamiltonians with a very small number of 2-qubit gates and trainable parameters.

5.2 Quantum chemistry

In this section, we utilize the recently introduced Curriculum Reinforcement Learning for Quantum Architecture Search (CRLQAS) [11] algorithm to automate the search for PQCs for quantum chemistry problems. Replacing the MLP in the curriculum reinforcement learning algorithm by KAN, we evaluate the performance of the neural network compared to an MLP. These neural networks are utilized to find the ground state of 4-qubit H_2 and LiH molecules. The molecular structures are discussed in detail in Appendix A.3 in the Table 1.

In finding the ground state of H_2 molecule, we consider an MLP of depth 6 (equivalently containing 5 hidden layers) with each layer containing 1000 neurons, i.e. $MLP_{6,1000}$, which is compared with a KAN of depth 4 (equivalently 3 hidden layers) with 3 neurons

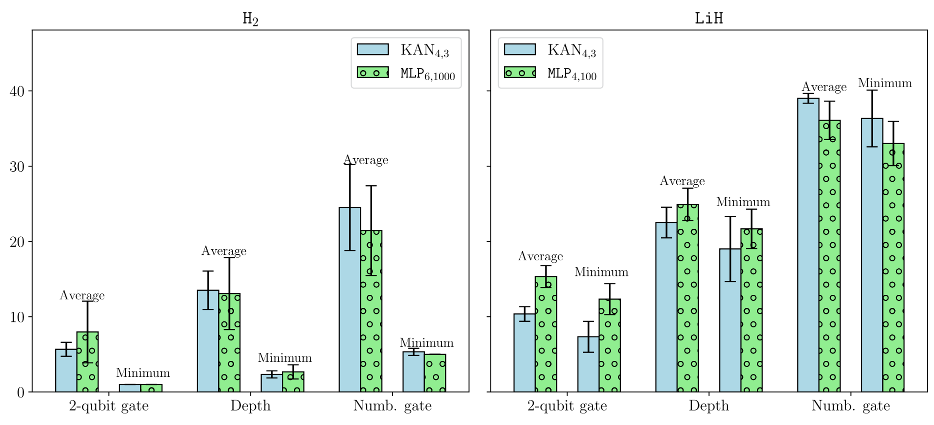


Figure 5 KAN outperforms MLP in finding a parameterized quantum circuit that solves the 4-qubit LiH and H_2 molecule. We solve a molecule when its energy goes below the chemical accuracy, i.e., 1.6×10^{-3} Hartree. In the plot, the average corresponds to the average performance of the neural networks over three distinct initializations, and the minimum is the best-performing seed among the three. We consider $\mathrm{KAN}_{4,3}$ that corresponds to a KAN of depth 4 and 3 neurons, whereas $\mathrm{MLP}_{6,1000}$ and $\mathrm{MLP}_{4,100}$ defines MLP of depth 6 with 1000 neurons and MLP of depth 4 with 100 neurons respectively. It should be noted that an MLP of configuration $\mathrm{MLP}_{6,1000}$ outperforms KAN in the number of 2-qubit gates, depth, and the total number of parameters, but the number of trainable parameters required for the MLP is 5.15×10^6 which is very large compared to 5.52×10^4 parameters of KAN

i.e. $\mathrm{MLP}_{4,3}$. Meanwhile, to find the ground state of LiH molecule we consider $\mathrm{MLP}_{4,100}$ and compare its performance with $\mathrm{KAN}_{4,3}$. For more details on the configuration of the neural networks see Table 6.

In Fig. 5, we benchmark the results by calculating the total number of 2-qubit gates, the depth, and the total number of gates in the PQC. These parameters are labelled as *2-qubit gate*, *Depth*, and *Numb. gate* on the x-axis of Fig. 5. In both cases, we observe that *$\mathrm{KAN}_{4,3}$ outperforms various MLP configurations in terms of average (as well as minimum) depth and 2-qubit gate count. This indicates that $\mathrm{KAN}_{4,3}$ can find more compact PQCs for solving the H_2 and LiH molecules compared to $\mathrm{MLP}_{6,1000}$ and $\mathrm{MLP}_{4,100}$*. However, the total number of gates required to solve these problems is smaller with MLP, suggesting that for a parameterized quantum circuit with fewer 1-qubit gates, it is preferable to choose MLP over KAN in CRLQAS.

In Appendix A.4, we further investigate two different configurations of KAN and MLP to find the ground state of the 4-qubit LiH. The results reveal that while larger configurations, such as $\mathrm{MLP}_{6,1000}$, outperform KAN, they require $5\times$ to $100\times$ more learnable parameters. However, when *considering a comparable number of learnable parameters, for instance, $\mathrm{MLP}_{4,500}$ and $\mathrm{KAN}_{4,50}$, KAN achieves the ground state with a shallower PQC and fewer 2-qubit gates*. Although MLP results in a lower error in estimating the ground state, our objective in quantum chemistry is to attain chemical accuracy, defined as 1.6×10^{-3} Hartree[2] [47]. *Given that the noise in 2-qubit gates is significantly higher than in 1-qubit gates [48], KAN demonstrates considerable promise for solving quantum chemistry problems on real quantum hardware.*

[2]The chemical accuracy is often defined as being within 1 kcal/mol, which is an experimental value, as 1 Hartree = 627.5095 kcal/mol, therefore 1 kcal/mol is equal to approximately $0.0015934 \approx 0.0016$ Hartree.

Table 3 KAN requires 2-3× less learnable parameters in the task of quantum state preparation. Meanwhile, in finding the ground state of $\mathtt{LiH}$ and $\mathtt{H}_2$ KAN takes 2.4-93× fewer parameters respectively and still solves the problems with smaller parameterized quantum circuits than MLP. To simplify, we denote the networks by their depth and number of neurons. For instance, a KAN with configuration [288, 3, 21] is abbreviated as $\text{KAN}_{2,3}$, where 2 represents the network depth (excluding the input layer) and 3 denotes the number of neurons in the hidden layer. Meanwhile, 288 is the size of the input calculated using the Eq. (9) for $D_{\text{max}} = 12$, $N = 3$, and $N_{1q} = 5$, and 21 is the number of actions. The same shorthand notation is applied to MLP networks

Problem	Network	Configuration	Spline and grid	Parameter count
Bell state prep. (noiseless)	KAN	$\text{KAN}_{2,2}$	B-spline, $k = 3$, $G = 5$	**1728**
	MLP	$\text{MLP}_{4,30}$	NA	4782
GHZ state prep. (noiseless)	KAN	$\text{KAN}_{2,3}$	B-spline, $k = 3$, $G = 5$	**5562**
	MLP	$\text{MLP}_{4,30}$	NA	11,181
Bell state prep. (noisy)	KAN	$\text{KAN}_{3,10}$	B-spline, $k = 4$, $G = 5$	**9540**
	MLP	$\text{MLP}_{4,100}$	NA	29,912
$\mathtt{H}_2$ ground state	KAN	$\text{KAN}_{4,3}$	B-spline, $k = 4$, $G = 5$	$\mathbf{5.525 \times 10^4}$
	MLP	$\text{MLP}_{6,1000}$	NA	5.150×10^6
$\mathtt{LiH}$ ground state	KAN	$\text{KAN}_{4,3}$	B-spline, $k = 4$, $G = 5$	$\mathbf{5.525 \times 10^4}$
	MLP	$\text{MLP}_{4,100}$	NA	1.348×10^5

Table 4 A new version of KAN [49] is approximately 3.46× slower than MLPs, which is far better than the older version, which was 120× slower for quantum state preparation. Meanwhile, KAN is 2.92× and 2.787× slower than MLP in finding the ground state of 4-qubit $\mathtt{H}_2$ and $\mathtt{LiH}$ molecule respectively

Problem	Network	Configuration	Avg. time per episode
GHZ state prep.	KAN	$\text{KAN}_{2,3}$	0.2049
	MLP	$\text{MLP}_{4,30}$	**0.0592**
$\mathtt{H}_2$ ground state	KAN	$\text{KAN}_{4,3}$	2.2103
	MLP	$\text{MLP}_{6,1000}$	**0.7552**
$\mathtt{LiH}$ ground state	KAN	$\text{KAN}_{4,3}$	3.3737
	MLP	$\text{MLP}_{4,100}$	**1.2102**

6 Resource details

Here we quantify the resources such as (1) the total number of learnable parameters and (2) the time of execution of each episode with the two networks required by KAN and MLP for the simulations presented in the previous section.

6.1 Parameter count

With depth L and width N, an MLP only requires $O(N^2L)$ parameters whereas a KAN requires $O(N^2L(G + k))$ parameters. In Table 3 for the quantum state preparation and in Table 7 for quantum chemistry problems, we calculate the total number of learnable parameters required for KANs and MLPs in noisy and noiseless scenarios. *We observe that in all cases KAN requires a lesser number of parameters than MLP.*

6.2 Time of executing each episode

Although KAN requires fewer learnable parameters than MLPs, the average time of executing each episode for KAN is 120× higher. *A recently released version of KAN, named MultKAN [49], executes an episode approximately* 3.46× *slower than MLPs*. The time required for KAN to solve state preparation and quantum chemistry problems are summarized in Table 4.

7 Discussion and future work

This work analyses the practicality of the question: *Can Kolmogorov-Arnold Network (KAN) replace the Multi-Layer Perceptron (MLP) structure in Quantum Architecture Search (QAS) to achieve comparable or better performance while mitigating the issues of parameter growth and activation function selection? We demonstrate that the answer is affirmative by tackling the quantum state preparation task and finding the ground state of chemical Hamiltonians*. To this end, we propose KAN for QAS, namely KANQAS, which replaces the MLP in the reinforcement learning-assisted quantum architecture search of the double deep Q-network with the KAN.

Our experiments reveal that *KANQAS can increase the probability of success of the RL-agent in finding optimal quantum circuits compared to MLPs in constructing multi-qubit maximally entangled states with non-parameterized gates*. Moreover, *when noise is present in the quantum device, achieving similar or better outcomes with an MLP necessitates a deeper and wider network compared to KAN, as well as a careful selection of the appropriate activation function. Since finding the optimal activation function for deep learning remains an ongoing challenge [50, 51], KAN has an advantage, as its activation functions are inherently learnable.*

In addressing quantum chemistry problems, KAN demonstrates the ability to provide a more compact parameterized quantum circuit, with less number of 2 qubit gate and depth for finding the ground state of the chemical Hamiltonian using VQE, with a significantly reduced number of learnable parameters. It is important to note that for larger configurations, such as $\text{MLP}_{6,1000}$, MLP outperforms KAN configurations but requires 5-100$\times$ more learnable parameters. When *considering a comparable number of learnable parameters, for example, $MLP_{4,500}$ and $KAN_{4,50}$, KAN achieves the ground state with a parameterized quantum circuit of smaller depth and fewer 2-qubit gates. Although MLP results in a lower error in estimating the ground state, our objective in quantum chemistry is to attain chemical accuracy, defined as* 1.6×10^{-3} *Hartree. Given that the noise in 2-qubit gates is significantly higher than in 1-qubit gates [48], KAN demonstrates considerable promise for solving quantum chemistry problems on real quantum hardware.*

Although the number of learnable parameters in KAN is on average 2-100$\times$ lesser than in MLPs, one of the biggest disadvantages of KAN is that it requires 2-3$\times$ more execution time per episode as compared to MLPs. However, due to their effectiveness and efficiency in finding solutions in noiseless and noisy quantum devices, KAN thus appears to be a reasonable alternative to traditional MLPs in solving quantum architecture search problems. In the following, we discuss the *future research directions* as a follow-up to our research.

- *KAN for VQAs*: A primary direction for future research is addressing the quantum architecture search problem within variational quantum algorithms, in this paper we just show the simulation of 4-qubit $\mathtt{H}_2$ and $\mathtt{LiH}$ molecules, this study should be expanded to bigger molecules such as $\mathtt{H}_2\mathtt{O}$ 8-qubits. These KAN-assisted algorithms could have significant applications in quantum chemistry and combinatorial optimization problems.
- *Optimizing computational time of KAN*: Another important goal is to explore the use of specialized accelerators, such as tensor processing units or digital signal processors, to reduce the computation time of KAN.
- *Interpretability of KAN*: Focusing on the interpretability of KAN, future research should investigate its applicability to similar but scalable problems to enhance

understanding. This can, for example, include the investigation of subclasses of families of the activation function after training KAN towards concept discovery [52]. *We invite readers to explore our Git repository [53] for inspiration and further motivation in this direction.*

Appendix

A.1 Tensor-based binary encoding of parametric quantum circuit

We use a binary encoding scheme [9, 11] that captures the structure of the Parametric Quantum Circuit (PQC), specifically the order of gates, to provide the agent with a full circuit description. To maintain a constant input size, the tensor is prepared for the deepest possible quantum circuit.

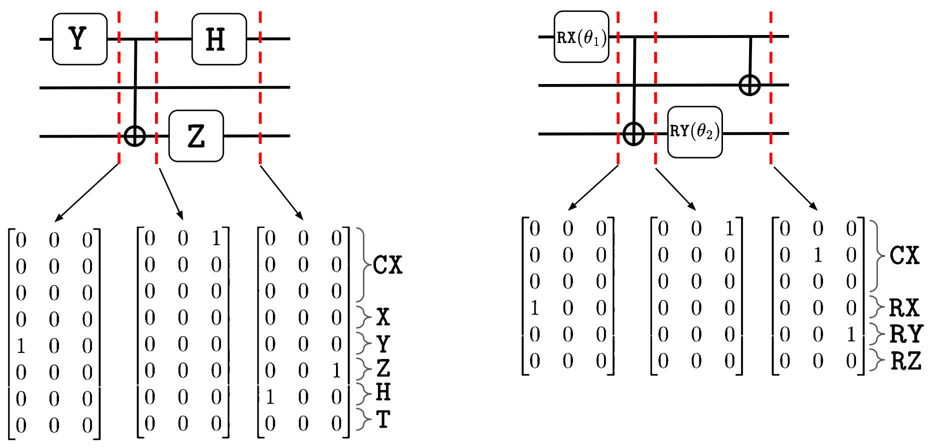


Figure 6 (a) For quantum state reconstruction the number of 1-qubit gates is one 5 so we encode the quantum circuit into a tensor of dimension $\left[D_{\max} \times ((N+5) \times N)\right]$ (b) whereas, for the quantum chemistry problem, the PQC is encoded into a tensor of dimension $\left[D_{\max} \times ((N+3) \times N)\right]$ due to the presence of 3 1-qubit gates

To construct the tensor, we first set the hyperparameter $D_{\max}$, which limits the maximum number of allowed gates (actions) in all episodes. A *moment* in a PQC refers to all gates that can be executed simultaneously, defining the circuit's depth.

We represent PQCs as 3D tensors where, at each episode, we initialize an empty circuit of depth $D_{\max}$, defined by a $\left[D_{\max} \times \left((N+N_{1q}) \times N\right)\right]$ tensor of zeros, where N is the number of qubits, and N_{1q} is the number of 1-qubit gates. Each matrix in the tensor has N rows for control and target qubit positions in `CNOT` gates, followed by either 3 (for quantum chemistry problems) or 5 (for quantum state reconstruction problems) rows indicating the positions of 1-qubit gates as represented in Fig. 6.

A.2 Double Deep Q-Network (DDQN)

Deep Reinforcement Learning (RL) techniques utilize Neural Networks (NNs) to adapt the agent's policy to optimize the return:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \tag{14}$$

where $\gamma \in [0,1)$ is the discount factor. For each state-action pair (s,a), an action value is assigned, quantifying the expected return from state s at step t when taking action a under policy π:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|s_t = s, a_t = a]. \tag{15}$$

The goal is to determine the optimal policy that maximizes the expected return, which can be derived from the optimal action-value function q_*, defined by the Bellman optimality equation:

$$q_*(s,a) = \mathbb{E}\left[r_{t+1} + \max_{a'} q_*(s_{t+1},a')|s_t = s, a_t = a\right]. \tag{16}$$

Instead of solving the Bellman optimality equation directly, value-based RL aims to learn the optimal action-value function from data samples. Q-learning is a prominent value-based RL algorithm, where each state-action pair (s,a) is assigned a Q-value $Q(s,a)$, which is updated to approximate q_*. Starting from randomly initialized values, the Q-values are updated according to the rule:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left(r_{t+1} + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)\right), \tag{17}$$

where α is the learning rate, r_{t+1} is the reward at time $t+1$, and s_{t+1} is the state encountered after taking action a_t in state s_t. This update rule is proven to converge to the optimal Q-values in the tabular case if all (s,a) pairs are visited infinitely often [54]. To ensure sufficient exploration in a Q-learning setting, an ϵ-greedy policy is used. Formally, the policy is defined as:

$$\pi(a|s) := \begin{cases} 1-\epsilon_t & \text{if } a = \max_{a'} Q(s,a'), \\ \epsilon_t & \text{otherwise.} \end{cases} \tag{18}$$

The ϵ-greedy policy introduces randomness to the actions during training, but after training, a deterministic policy is used.

We employ NN ad function approximations to extend Q-learning to large state and action spaces. NN training typically requires independently and identically distributed data. This problem is circumvented by experience replay. This method divides experiences into single-episode updates and creates batches which are randomly sampled from memory. For stabilizing training, two NNs are used: a policy network, which is continuously updated, and a target network, which is an earlier copy of the policy network. The current value is estimated by the policy network, while the target network provides a more stable target value Y given by:

$$Y_{\mathrm{DQN}} = r_{t+1} + \gamma \max_{a'} Q_{\mathrm{target}}(s_{t+1}, a'). \tag{19}$$

In the DDQN algorithm, we sample the action for the target value from the policy network to reduce the overestimation bias present in standard DQN. The corresponding target is defined as:

$$Y_{\mathrm{DDQN}} = r_{t+1} + \gamma Q_{\mathrm{target}}\left(s_{t+1}, \arg\max_{a'} Q_{\mathrm{policy}}(s_{t+1}, a')\right). \tag{20}$$

This target value is approximated via a loss function. In this work, we consider the loss function as the smooth L1-norm.

A.3 Configuration of neural networks

The detailed configurations of the KANs and the MLPs utilized in the main text to tackle quantum state preparation and quantum chemistry problems are provided in Table 5 and Table 6

Table 5 Configuration for noiseless GHZ and Bell state preparation. In the case of MLP, the activation function is fixed whereas it is learned during the network training for KAN. For the sake of simplicity, we represent the networks by their depth and number of neurons for example, for a KAN of configuration [288, 3, 21] we use the shorthand notation $\mathrm{KAN}_{2,3}$ as the KAN is of depth 2 (i.e. 1 hidden layer) and contains 3 neurons in the hidden layer. The KAN takes 288 size input which is the size of the tensor that encodes the quantum state calculated using Eq. (9) for $D_{\max} = 12$, $N = 3$, and $N_{1q} = 5$, and returns 21 actions

Network	Problem	Configuration	Spline and grid	Activation func.
KAN	Bell state prep.	$\mathrm{KAN}_{2,2}$	B-spline, $k = 3$, $G = 5$	learnable
	GHZ state prep.	$\mathrm{KAN}_{2,3}$	B-spline, $k = 4$, $G = 5$	learnable
MLP	Bell state prep.	$\mathrm{MLP}_{4,30}$	NA	LeakyReLU
	GHZ state prep.	$\mathrm{KAN}_{2,3}$		
	GHZ state prep.	$\mathrm{KAN}_{4,30}$		

Table 6 The configuration of KAN and MLP utilized in tackling quantum chemistry problems specifically for finding the ground state of 4-qubit $\mathtt{H}_2$ and $\mathtt{LiH}$ molecule. A detailed configuration of these molecules is in Tab. 1. As discussed in the caption of Tab. 5, we represent the networks by their depth and number of neurons for example, for an MLP of configuration [1121, 500, 500, 500, 24], we use the shorthand notation $\mathrm{MLP}_{4,500}$ as the MLP of depth 4 (excluding the input layer) and of 500 neurons in the hidden layer

Network	Molecule	Network configuration	Spline and grid	Activation func.
KAN	$\mathtt{H}_2$	$\mathrm{KAN}_{4,3}$	B-spline, $k = 10$, $G = 5$	learnable
	$\mathtt{LiH}$	$\mathrm{KAN}_{2,3}$		
		$\mathrm{KAN}_{4,30}$		
		$\mathrm{KAN}_{4,50}$		
MLP	$\mathtt{H}_2$	$\mathrm{MLP}_{6,1000}$	NA	LeakyReLU
	$\mathtt{LiH}$	$\mathrm{MLP}_{4,100}$		
		$\mathrm{MLP}_{4,500}$		
		$\mathrm{MLP}_{6,1000}$		

Table 7 We observe that although $\text{MLP}_{6,1000}$ gives the best results in all benchmarking variables, it requires 10× more learnable parameters than $\text{KAN}_{4,30}$, making it significantly more expensive in terms of training time. For a fair comparison, $\text{KAN}_{4,50}$ and $\text{MLP}_{4,500}$ show a tradeoff between the number of parameterized 1-qubit gates and 2-qubit gates, with $\text{MLP}_{4,500}$ having 1.078 times more parameters than $\text{KAN}_{4,50}$. Despite having fewer parameters, KAN provides a competitive approximation of the ground state with a shorter parameterized quantum circuit, fewer 2-qubit gates, and smaller depth than MLP

Molecule	Network configuration	Min. 2-qubit gate	Min. depth	Min. 1-qubit gate	Error	Parameter count
4-qubit	$\text{KAN}_{4,30}$	11	18	15	1.129×10^{-4}	$\mathbf{5.784 \times 10^{5}}$
`LiH`	$\text{KAN}_{4,50}$	8	17	24	2.265×10^{-4}	9.960×10^{5}
	$\text{MLP}_{4,500}$	13	18	13	1.100×10^{-4}	1.074×10^{6}
	$\text{MLP}_{6,1000}$	**6**	**11**	**13**	$\mathbf{3.606 \times 10^{-5}}$	5.152×10^{6}

A.4 Quantum chemistry with different configurations of KAN and MLP

In this section, we further explore the different configurations of KAN and MLP in finding the ground state of the 4-qubit `LiH` molecule. We use the following variables to benchmark the performance of the different neural networks: the minimum number of 1-qubit gates (Min. 1-qubit gate), the minimum number of 2-qubit gates (Min 2-qubit gate), the error in estimating the ground state energy (error), and the number of learnable parameters in the network (Parameter count). The results are presented in detail in the Table 7.

From the Table 7 we observe although the $\text{MLP}_{6,1000}$ provides us with the best result in terms of all the benchmarking variables, the number of learnable parameters it requires to achieve this outcome is 10× higher than $\text{KAN}_{4,30}$, which is far more expensive in terms of training time. Meanwhile, for the sake of fair comparison, if we observe the performance of $\text{KAN}_{4,50}$ and $\text{MLP}_{4,500}$ where the number of parameters in MLP is 1.078× more than KAN, we notice a tradeoff between the number of parameterized 1-qubit gates and the number of 2-qubit gates. The KAN provides us with a competitive approximation of the ground state with a shorter parameterized quantum circuit, containing less 2-qubit gate and smaller depth than MLP despite having a smaller number of parameters. Considering that the noise in 2-qubit gates in real quantum hardware is many scales higher than 1-qubit gates, KAN with a comparable number of parameters, shows a large potential in quantum architecture search in quantum hardware.

List of Abbreviations

KANQAS, Kolmogorov-Arnold Network for Quantum Architecture Search; KAN, Kolmogorov-Arnold Network; MLP, Multi-Layer Perceptron; QAS, Quantum Architecture Search; CRL, Curriculum Reinforcement Learning; RL, Reinforcement Learning; VQA, Variational Quantum Algorithms; NN, Neural Network; NISQ, Noisy Intermediate-Scale Quantum; DDQN, Double Deep Q-Network; CRLQAS, Curriculum Reinforcement Learning for Quantum Architecture Search; GHZ, Green-berger–Horne–Zeilinger; VQE, Variational Quantum Eigensolver; PQC, Parameterized Quantum Circuit.

Acknowledgements

AK would like to thank Ziming Liu for fruitful discussions about KAN. AK wish to thank the Finnish Computing Competence Infrastructure (FCCI) for supporting this project with computational and data storage resources.

Author contributions

A.K. conceptualized the study, supervised the project, wrote the code, analyzed the results, and prepared the manuscript. A.S. analyzed the results and contributed to manuscript preparation. A. Sadhu analyzed and prepared the results, and participated in manuscript preparation.

Funding

Open Access funding provided by University of Helsinki (including Helsinki University Central Hospital). A.K. acknowledges funding from the Research Council of Finland through the Finnish Quantum Flagship project 358878 (UH). Open access funded by Helsinki University Library.

Data Availability

No datasets were generated or analysed during the current study.

Declarations

Ethics approval and consent to participate
Ethical approval for this study was obtained from the authors, and all the authors provided informed consent to participate.

Consent for publication
The authors grant permission for the publication of this research.

Competing interests
The authors declare no competing interests.

Author details
[1]QTF Centre of Excellence, Department of Physics, University of Helsinki, Helsinki, Finland. [2]Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Gliwice, Poland. [3]Joint Doctoral School, Silesian University of Technology, Gliwice, Poland. [4]Quantum Intelligence Alliance, Kolkata, India. [5]QuTech, Delft University of Technology, Delft, The Netherlands. [6]Raman Research Institute, Bengaluru, India. [7]Centre for Quantum Science and Technology (CQST), International Institute of Information Technology, Hyderabad, Telangana, India.

Received: 2 August 2024 Accepted: 4 November 2024 Published online: 12 November 2024

References
1. Ren P, Xiao Y, Chang X, Huang P-Y, Li Z, Chen X, Wang X. A comprehensive survey of neural architecture search: challenges and solutions. ACM Comput Surv. 2021;54(4):1–34.
2. Zhang S-X, Hsieh C-Y, Zhang S, Yao H. Differentiable quantum architecture search. Quantum Sci Technol. 2022;7(4):045023.
3. Lu Z, Shen P-X, Deng D-L. Markovian quantum neuroevolution for machine learning. Phys Rev Appl. 2021;16(4):044039.
4. McClean JR, Romero J, Babbush R, Aspuru-Guzik A. The theory of variational hybrid quantum-classical algorithms. New J Phys. 2016;18(2):023023.
5. Cerezo M, Arrasmith A, Babbush R, Benjamin SC, Endo S, Fujii K, McClean JR, Mitarai K, Yuan X, Cincio L, et al. Variational quantum algorithms. Nat Rev Phys. 2021;3(9):625–44.
6. Sarkar A. Automated quantum software engineering. Autom Softw Eng. 2024;31(1):1–17.
7. Kuo E-J, Fang Y-LL, Chen SY-C. Quantum architecture search via deep reinforcement learning. ArXiv preprint. arXiv:2104.07715 (2021).
8. Ostaszewski M, Trenkwalder LM, Masarczyk W, Scerri E, Dunjko V. Reinforcement learning for optimization of variational quantum circuit architectures. Adv Neural Inf Process Syst. 2021;34:18182–94.
9. Kundu A, Bedełek P, Ostaszewski M, Danaci O, Patel YJ, Dunjko V, Miszczak JA. Enhancing variational quantum state diagonalization using reinforcement learning techniques. New J Phys. 2024;26(1):013034.
10. Du Y, Huang T, You S, Hsieh M-H, Tao D. Quantum circuit architecture search for variational quantum algorithms. npj Quantum Inf. 2022;8(1):62.
11. Patel YJ, Kundu A, Ostaszewski M, Bonet-Monroig X, Dunjko V, Danaci O. Curriculum reinforcement learning for quantum architecture search under hardware errors. ArXiv preprint. arXiv:2402.03500 (2024).
12. Sadhu A, Sarkar A, Kundu A. A quantum information theoretic analysis of reinforcement learning-assisted quantum architecture search. Quantum Mach Intell. 2024;6(2):49. https://doi.org/10.1007/s42484-024-00181-0.
13. Morgillo AR, Mangini S, Piastra M, Macchiavello C. Quantum state reconstruction in a noisy environment via deep learning. Quantum Mach Intell. 2024;6(2):39. https://doi.org/10.1007/s42484-024-00168-x.
14. Lumino A, Polino E, Rab AS, Milani G, Spagnolo N, Wiebe N, Sciarrino F. Experimental phase estimation enhanced by machine learning. Phys Rev Appl. 2018;10(4):044033.
15. Wang S, Fontana E, Cerezo M, Sharma K, Sone A, Cincio L, Coles PJ. Noise-induced barren plateaus in variational quantum algorithms. Nat Commun. 2021;12(1):6961.
16. Bittel L, Kliesch M. Training variational quantum algorithms is np-hard. Phys Rev Lett. 2021;127:120502. Available: https://link.aps.org/doi/10.1103/PhysRevLett.127.120502.
17. Liu Z, Wang Y, Vaidya S, Ruehle F, Halverson J, Soljačić M, Hou TY, Tegmark M. Kan: Kolmogorov-Arnold networks. ArXiv preprint. arXiv:2404.19756 (2024).
18. Aghaei AA. fkan: fractional Kolmogorov-Arnold networks with trainable Jacobi basis functions. ArXiv preprint. arXiv:2406.07456 (2024).
19. Genet R, Inzirillo H. Tkan: temporal Kolmogorov-Arnold networks. ArXiv preprint. arXiv:2405.07344 (2024).
20. Bozorgasl Z, Chen H. Wav-kan: wavelet Kolmogorov-Arnold networks. ArXiv preprint. arXiv:2405.12832 (2024).
21. Abueidda DW, Pantidis P, Mobasher ME. Deepokan: deep operator network based on Kolmogorov Arnold networks for mechanics problems. ArXiv preprint. arXiv:2405.19143 (2024).
22. Kiamari M, Kiamari M, Krishnamachari B. Gkan: graph Kolmogorov-Arnold networks. ArXiv preprint. arXiv:2406.06470 (2024).
23. Xu J, Chen Z, Li J, Yang S, Wang W, Hu X, Ngai EC-H. Fourierkan-gcf: fourier Kolmogorov-Arnold network–an effective and efficient feature transformation for graph collaborative filtering. ArXiv preprint. arXiv:2406.01034 (2024).
24. Genet R, Inzirillo H. A temporal Kolmogorov-Arnold transformer for time series forecasting. ArXiv preprint. arXiv:2406.02486 (2024).
25. Xu K, Chen L, Wang S. Kolmogorov-Arnold networks for time series: bridging predictive power and interpretability. ArXiv preprint. arXiv:2406.02496 (2024).
26. Vaca-Rubio CJ, Blanco L, Pereira R, Caus M. Kolmogorov-Arnold networks (kans) for time series analysis. ArXiv preprint. arXiv:2405.08790 (2024).
27. Cheon M. Kolmogorov-Arnold network for satellite image classification in remote sensing. ArXiv preprint. arXiv:2406.00600 (2024).

28. Wang Y, Sun J, Bai J, Anitescu C, Eshaghi MS, Zhuang X, Rabczuk T, Liu Y. Kolmogorov Arnold informed neural network: a physics-informed deep learning framework for solving pdes based on Kolmogorov Arnold networks. ArXiv preprint. arXiv:2406.11045 (2024).
29. Zhang S-X, Hsieh C-Y, Zhang S, Yao H. Neural predictor based quantum architecture search. Mach Learn: Sci Technol. 2021;2(4):045027.
30. Wu W, Yan G, Lu X, Pan K, Yan J. Quantumdarts: differentiable quantum architecture search for variational quantum algorithms. In: International conference on machine learning. PMLR; 2023. p. 37745–64.
31. Wang H, Ding Y, Gu J, Lin Y, Pan DZ, Chong FT, Han S. Quantumnas: noise-adaptive search for robust quantum circuits. In: 2022 IEEE international symposium on High-Performance Computer Architecture (HPCA). IEEE; 2022. p. 692–708.
32. He Z, Wei J, Chen C, Huang Z, Situ H, Li L. Gradient-based optimization for quantum architecture search. Neural Netw. 2024;179:106508.
33. He Z, Deng M, Zheng S, Li L, Situ H. Training-free quantum architecture search. In: Proceedings of the AAAI conference on artificial intelligence. vol. 38. 2024. p. 12430–8.
34. Situ H, He Z, Zheng S, Li L. Distributed quantum architecture search. Phys Rev A. 2024;110(2):022403.
35. Patel YJ, Jerbi S, Bäck T, Dunjko V. Reinforcement learning assisted recursive qaoa. EPJ Quantum Technol. 2024;11(1):6.
36. Zhu W, Pi J, Peng Q. A brief survey of quantum architecture search. In: Proceedings of the 6th international conference on algorithms, computing and systems. 2022. p. 1–5.
37. Martyniuk D, Jung J, Paschke A. Quantum architecture search: a survey. ArXiv preprint. arXiv:2406.06210 (2024).
38. Kundu A. Reinforcement learning-assisted quantum architecture search for variational quantum algorithms. ArXiv preprint. arXiv:2402.13754 (2024).
39. Levine IN, Busch DH, Shull H. Quantum chemistry. vol. 6. Pearson Prentice Hall Upper Saddle River; 2009.
40. Kolmogorov AN. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. In: Doklady Akademii Nauk. vol. 114. Russian Academy of Sciences; 1957. p. 953–6.
41. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. Neural Netw. 1989;2(5):359–66.
42. Knott GD. Interpolating cubic splines. vol. 18. Berlin: Springer; 1999.
43. Rumelhart DE, Durbin R, Golden R, Chauvin Y. Backpropagation: the basic theory. In: Backpropagation. Psychology Press; 2013. p. 1–34.
44. Waris R, Corentin. Kolmogorov-Arnold Q-Network (KAQN)-KAN applied to Reinforcement learning, initial experiments. 2024. https://github.com/riiswa/kanrl.
45. Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 30. 2016.
46. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, et al. Human-level control through deep reinforcement learning. Nature. 2015;518(7540):529–33.
47. Chen Y, Zhang L, Wang H, E W. Ground state energy functional with Hartree–Fock efficiency and chemical accuracy. J Phys Chem A. 2020;124(35):7155–65.
48. Wilson E, Mueller F, Bassman Oftelie L, Iancu C. Empirical evaluation of circuit approximations on noisy quantum devices. In: Proceedings of the international conference for high performance computing, networking, storage and analysis. 2021. p. 1–15.
49. MultKAN. 2024. https://github.com/KindXiaoming/pykan/blob/master/kan/MultKAN.py.
50. Hayou S, Doucet A, Rousseau J. On the selection of initialization and activation function for deep neural networks. ArXiv preprint. arXiv:1805.08266 (2018).
51. Ramachandran P, Zoph B, Le QV. Searching for activation functions. ArXiv preprint. arXiv:1710.05941 (2017).
52. Sarkar A, Kundu A, Steinberg M, Mishra S, Fauquenot S, Acharya T, Miszczak JA, Feld S. Yaqq: yet another quantum quantizer–design space exploration of quantum gate sets using novelty search. ArXiv preprint. arXiv:2406.17610 (2024).
53. Kundu A. KANQAS GitHub. 2024. https://github.com/Aqasch/KANQAS_code.
54. Melo FS. Convergence of q-learning: a simple proof. Institute of Systems and Robotics, Tech Rep.; 2001. p. 1–4.

Publisher's Note