Quantum circuit architecture search: error mitigation and trainability enhancement for variational quantum solvers

Yuxuan Du,¹ Tao Huang,² Shan You,² Min-Hsiu Hsieh,³ and Dacheng Tao¹

¹UBTECH Sydney AI Centre, School of Computer Science, Faculty of Engineering, The University of Sydney, Australia
²SenseTime
³Centre for Quantum Software and Information, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia

(Dated: November 13, 2020)

Quantum error mitigation techniques are at the heart of quantum hardware implementation, and are the key to performance improvement of the variational quantum learning scheme (VQLS). Although VQLS is partially robust to noise, both empirical and theoretical results exhibit that noise would rapidly deteriorate the performance of most variational quantum algorithms in large-scale problems. Furthermore, VQLS suffers from the barren plateau phenomenon—the gradient generated by the classical optimizer vanishes exponentially with respect to the qubit number. Here we devise a resource and runtime efficient scheme, the quantum architecture search scheme (QAS), to maximally improve the robustness and trainability of VQLS. In particular, given a learning task, QAS actively seeks an optimal circuit architecture to balance benefits and side-effects brought by adding more quantum gates. Specifically, while more quantum gates enable a stronger expressive power of the quantum model, they introduce a larger amount of noise and a more serious barren plateau scenario. Consequently, QAS can effectively suppress the influence of quantum noise and barren plateaus. We implement QAS on both the numerical simulator and real quantum hardware, via the IBM cloud, to accomplish data classification and quantum chemistry tasks. Numerical and experimental results show that QAS significantly outperforms conventional variational quantum algorithms with heuristic circuit architectures. Our work provides practical guidance for developing advanced learning-based quantum error mitigation techniques on near-term quantum devices.

I. Introduction

Quantum error mitigation is an indispensable technique to realize large-scale quantum computation [1, 2]. During the past decades, a considerable effort has been dedicated to devising various quantum error correction codes such as surface codes and toric codes, to detect and correct errors caused by decoherence and imperfect control [3–6]. However, the implementation of these quantum error correction protocols requests expensive quantum resources, including huge ancillary qubits and multi-control quantum gates, which are prohibitive to noisy intermediate scale quantum (NISQ) devices [2]. For example, to fabricate a logical qubit whose physical error rate is below 1% threshold, $10^3−10^4$ physical qubits are required [7]. To this end, resource-efficient alternatives have been widely explored to mitigate errors in NISQ machines. A representative method to suppress the noise effect is quasi-probability strategies [8, 9]. Specifically, a linear combination of noisy circuits with suitable coefficients can achieve probabilistic error cancellation, attributing to the linearity of quantum mechanics. If the error distribution is known in advance, a noiseless quantum circuit could be returned. Although this method is resource-efficient without ancillary qubits, its scalability is limited since capturing the error distribution of a large quantum system is intractable. In addition to the quasi-probability approach, other error mitigation techniques, e.g., extrapolation [10], quantum subspace expansion [11], and data-driven methods [12, 13], have been proposed. Nevertheless, the runtime of these methods is not affordable. A general quantum error mitigation technique that is both runtime and resource-efficient remains largely unknown.

The variational quantum learning scheme (VQLS) [14], which is also known as the quantum neural network [15, 17], is one of the most promising candidates to use NISQ devices to solve practical tasks that are beyond the reach of classical computers. Given an input $z$ and an objective function $\mathcal{L}$, VQLS employs a gradient-based classical optimizer that continuously updates parameters in a trainable quantum circuit $U(\theta)$ to find the optimal $\theta^*$, i.e.,

$$
\theta^* = \arg \min_{\theta \in \mathcal{C}} \mathcal{L}(\theta, z),
$$

where $\mathcal{C} \subseteq \mathbb{R}^d$ is a constraint set, and $\theta$ are adjustable parameters of quantum gates [18, 19]. For instance, when VQLS is specified as an Eigen-solver [14], $z$ refers to a Hamiltonian and the objection function could be chosen as $\mathcal{L} = \text{Tr}(z |\psi(\theta)\rangle\langle\psi(\theta)|)$, where $|\psi(\theta)\rangle$ is the quantum state generated by $U(\theta)$. Similar to deep neural networks [20], the trainable quantum circuit $U(\theta)$ generally adopts a multi-layer layout,

$$
U(\theta) = \prod_{l=1}^{L} U_l(\theta),
$$

where $U_l(\theta)$ consists of a sequence of parameterized single qubit and two qubits quantum gates, and $L$ denotes the circuit depth. Note that the arrangement of quantum gates in $U_l(\theta)$ is flexible, enabling VQLS to adequately...
use available quantum resources and to accord with any physical restriction.

The effectiveness of VQLS towards small-scale learning problems has been validated by experimental studies. For example, a VQLS based algorithm has been carried out on a two qubits quantum processor to classify a synthetic dataset with high performance [16]. Such a result indicates that for small circuit depth $L$ and qubit number $N$, VQLS can well approximate the target result $\theta^*$, even if the information received by the classical optimizer is slightly perturbed. However, the performance of VQLS will degrade significantly when $N$ and $L$ become large, due to the following two reasons. First, VQLS suffers from the barren plateau phenomenon, where the gradient can vanish exponentially with respect to the increased circuit depth and the qubits count [21] [22]. Namely, the hardness to train VQLS to find $\theta^*$ is exponentially increased in terms of $N$ and $L$. Second, VQLS still suffers from quantum system noise, where their performance can be sharply varied under noiseless and noisy settings [23]. In addition, theoretical studies demonstrated that the amount of noise contained in the gradient information is polynomially scaled with $N$ and $L$, which may render the divergent optimization and exacerbates the barren plateau phenomenon [21] [23]. As a result, efficient error mitigation techniques become an indispensable ingredient to enable VQLS based algorithms to solve intermediate-scale problems with guaranteed performance. A concrete example is applying a variational Eigen-solver associated with advanced error mitigation methods to accurately model the binding energy of the hydrogen chains up to a dozen qubits on the Sycamore quantum processor [20]. Therefore, it is highly desired to propose a general scheme for VQLS that has the ability to simultaneously inhibit the noise affect (i.e., error mitigation) and to alleviate the barren plateau phenomenon (trainability enhancement).

Achieving error mitigation and trainability enhancement for VQLS is challenging. The fundamental difficulty is that under the NISQ setting, more quantum resources are required for the trainable quantum circuits $U(\theta)$ resulting in both a positive and negative aftermath. In particular, the expressive power of $U(\theta)$ will be strengthened by increasing the number of layers $L$ because the hypothesis class represented by $U(\theta)$ is enlarged [18] [19]. Meanwhile, a deep circuit depth implies that the gradient information received by the classical optimizer vanishes exponentially and is full of noise, which may lead to barren plateaus and divergent optimization. Consequently, the error mitigation and the trainability enhancement in VQLS result in seeking a circuit architecture that utilizes the least quantum resources to implement $U(\theta)$ that cover the target concept. Initial studies [27] [28] have exploited genetic algorithms and quantum gates pruning methods to achieve this goal. However, locating such an architecture is computationally hard, since the set of possible circuit architectures scales exponentially with respect to the qubits count and the circuit depth. Furthermore, quantum gates are accompanied by variable noise levels, e.g., two qubits gates are much noisier than single qubit gates. Incorporating such weighted noise further exacerbates the difficulty of the error mitigation design for VQLS. Last, the proposed method should be highly compatible with physical restrictions, since the qubits connectivity and the types of quantum gates available for different quantum devices are not unified.

To address all aforementioned concerns, we devise a quantum architecture search scheme (QAS) to accomplish the error mitigation and the trainability enhancement under the VQLS setting. The advantage of QAS is ensured by recasting the error mitigation and alleviation of the barren plateau in VQLS as a learning problem. Notably, QAS does not request any ancillary quantum resource, and its runtime is almost the same as conventional VQLS based algorithms. Moreover, QAS is compatible with all quantum platforms, e.g., optical, trapped-ion, and superconducting quantum machines, since it can actively adapt to physical restrictions and weighted noise of varied quantum gates. Moreover, QAS can seamlessly integrate with other quantum error mitigation methods and barren plateau solutions [8] [10] [22]. Celebrated by the universality and efficacy, QAS contributes to a broad class of variational quantum algorithms built on various quantum machines. To the best of our knowledge, this is the first proposal that can directly apply to current quantum machines to achieve the error mitigation and the trainability enhancement with scalability. Remarkably, QAS can be applied to other tasks beyond VQLS, which includes exploring protocols of the optimal quantum control and the approximation of the target unitary using the limited quantum gates.

II. The quantum architecture search

Let us formalize the error mitigation and trainability enhancement for VQLS as a learning task. Denote $\mathcal{S}$ as the set that contains all possible circuit architectures (or equivalent sequences of quantum gates) to build $U(\theta)$ in Eqn. (2). Notably, the cardinality of $\mathcal{S}$ is determined by the qubits count $N$, the maximum circuit depth $L$, and the number of allowed types of quantum gates $Q$, i.e., $|\mathcal{S}| = O(Q^{NL})$. The performance of VQLS heavily depends on the employed circuit architecture $a \sim \mathcal{S}$. Suppose the quantum system noise, induced by $a$, is modeled by the quantum channel $\mathcal{E}_a$. Taking into account of the circuit architecture information and the related noise, the objective of VQLS in Eqn. (1) can be rewritten as

$$\left(\theta^*, a^*\right) = \arg \min_{\theta \in \mathcal{C}, a \in \mathcal{S}} \mathcal{L}(\theta, a, z, \mathcal{E}_a).$$

(3)

The learning problem formulated in Eqn. (3) forces to output the best quantum circuit architecture $a^*$ to suppress the effect of noise and to alleviate the barren plateau influence. Despite the theoretical feasibility, the optimization of Eqn. (3) is difficult since the size of $\mathcal{S}$ grows exponentially.
Step 1. Supernet setup

The classical analog of the learning problem in Eqn. (3) is the neural network architecture search [29]. Recall that the success of deep learning is largely attributed to novel neural architectures for specific learning tasks, e.g., the convolutional neural networks for image processing tasks [29]. However, deep neural networks designed by human experts are generally time-consuming and error-prone [29]. To tackle the aforementioned issues, the neural architecture search approach, i.e., the process of automating architecture engineering, has been widely explored, and achieved state of the art performances in many learning tasks [30–33]. Despite having a similar aim, naively initializing independently, the total number of parameters for $A$ is $Q^NL$ with $Q$ being the number of types of trainable gates, e.g., $Q = 3$ (i.e., $R_X$, $R_Y$, and $R_Z$) in Step 1 of Figure 1. Such a parameter space is hard to optimize for a large $N$ and $L$. To tackle this issue, in QAS, $A$ adopts the weight sharing strategy to initialize the trainable parameters. Concretely, if the single-qubit gates of the $l$-th layer of some subnets share the same layout, then the training parameters of $U_l(\theta)$ for these subnets are same, regardless of variations in the layout of other layers (see Appendix A for more details). In this way, the number of parameters to be optimized in $A$ scales with $O(LQ^N)$, which is exponentially reduced in terms

The classical paradigm of the quantum architecture search scheme (QAS). In Step 1, QAS sets up the supernet $\mathcal{A}$. Supernet $\mathcal{A}$ collects all possible subnets $\mathcal{S} = \{a\}$ (i.e., variational quantum circuit architectures), where each subnet $a$ refers to a specific gates sequence to compose $U(\theta) = \prod_{i=N}^{1} U_i(\theta)$ in Eqn. (2). For example, one possible subnet $a$ shown in the figure is that $U_i(\theta)$ for all $i \in [L]$ obeys the same layout and $U_i(\theta) = (\otimes_{j=1}^{N} R_Z(\theta_j)) (\otimes_{j=1}^{N} R_Y(\theta_j)) T^{\otimes N}$, where all single qubit gates are $R_Z$ gates ($R_Y$ and $T$ gates) highlighted by the blue hexagons (the green and orange hexagons), and two qubits gates highlighted by the brown rectangle are identical. The Supernet $\mathcal{A}$ employs a weight sharing strategy to initialize trainable parameters in all subnets (see Appendix A for details). Specifically, if the layout of single qubit gates composed of $U_i(\theta)$ for any two subnets $a$ and $a'$ is identical, then these two subnets share the same training parameters for $U_i(\theta)$. The unitary $U_a$ refers to the encoding layer, e.g., loading classical inputs into the quantum states for machine learning tasks. In Step 2, given the specified learning task $L$, QAS iteratively samples a subnet $a^{(t)} \in \mathcal{S}$ and optimizes its trainable parameters to minimize $L$. After $T$ iterations, QAS moves to Step 3 and exploits the trained parameters $\theta^{(T)}$ and the predefined $L$ to compare the performances among $K$ subnets. The subnets are uniformly sampled from $\mathcal{S}$, where the subnet with the best performance is selected as the output, indicated by a red smiley face. Last, in Step 4, QAS utilizes the searched subnet and the parameters $\theta^{(T)}$ to retrain the quantum solver with few iterations.
of $L$. Once the initialization of the supernet is completed, QAS repeatedly executes the following procedures with $T$ iterations. As exhibited in Step 2 of Figure 1, the $t$-th iteration, QAS uniformly samples a subnet $\mathbf{a}^{(t)} \in S$, to minimize $\mathcal{L}$ in Eqn. (3), i.e., the parameters in $\mathbf{a}^{(t)}$ is updated to $\theta^{(t+1)} = \theta^{(t)} - \eta \partial \mathcal{L}(\theta^{(t)}, \mathbf{a}^{(t)}, z, \mathcal{E}_a) / \partial \theta^{(t)}$, with $\eta$ being the learning rate. After training, as depicted in Step 3 of Figure 1, QAS uniformly samples $K$ subnets, then ranks their performances, and assigns the subnet with the best performance as the output to approximate $\mathbf{a}^\ast$. Mathematically, denoted $K$ as the set collecting the sampled $K$ subnets, the output subnet yields

$$\arg \min_{\mathbf{a} \in K} \mathcal{L}(\theta^{(T)}, \mathbf{a}, z, \mathcal{E}_a).$$

Last, QAS employs the trained parameters $\theta^{(T)}$ to retrain the output subnet, shown in Step 4 of Figure 1.

We empirically observe fierce competition among different subnets in optimizing QAS (See Appendix C for details). This phenomenon is in contrast with the deep neural network architecture search. Namely, suppose all supernets has a close relationship the adversarial bandit problem follows the proposal of the quantum kernel classifier [34]. Moreover, the following theorem shows that the strategy used in QAS outperforms all bandit algorithms in terms of the regret measure.

**Theorem 1.** Let $W$ and $T$ be the number of supernets and iterations, respectively. Suppose that the subnet $\mathbf{a}^{(t)}$ is assigned to the $I_w^{(t)}$-th supernet $\mathcal{A}(I_w^{(t)})$ with $I_w^{(t)} \in [W]$ at the $t$-th iteration, where the corresponding objective function in Eqn. (3) is $\mathcal{L}(\theta^{(t)}, I_w^{(t)}, \mathbf{a}^{(t)}) \in [0, 1]$. Define the regret as

$$R_T = \sum_{t=1}^{T} \mathcal{L}(\mathbf{a}^{(t)}, I_w^{(t)}) - \min_{(w_t, i) \in \pi} \sum_{t=1}^{T} \mathcal{L}(\mathbf{a}^{(t), w_t}),$$

where the randomness is over the selection of $I_w^{(t)}$. The method used in QAS to determine $\{I_w^{(t)}\}$ promises the regret $R_T \leq 0$, while the regret for the best bandit algorithms is lower bounded by $R_T = \Omega(T)$.

The proof of Theorem 1 and the discussion about how to exploit bandit techniques to reduce the runtime in determining $\{I_w^{(t)}\}_{t=1}^{T}$ are provided in Appendix B.

### III. Experimental results

The proposed QAS is universal and facilitates a wide range of VQLS based learning tasks, e.g., machine learning [16, 35, 36], quantum chemistry [14, 20], and quantum information processing [27, 28]. In the following, we separately apply QAS to accomplish a classification task and a variational quantum Eigen-solver task to confirm its capability towards the error mitigation and the trainability enhancement. All numerical simulations are implemented in Python in conjunction with the PennyLane and the Qiskit packages [39, 40]. Specifically, PennyLane is the backbone to implement QAS and Qiskit supports different types of noisy models. The Python code related to the examples presented in this study will be available on Github [41]. We defer the explanation of basic terminologies in machine learning and quantum chemistry in Appendices C and D.

Here we first apply QAS to achieve a binary classification task under both the noiseless and noisy scenarios. Denote $D$ as the synthetic dataset, where its construction rule follows the proposal of the quantum kernel classifier [19]. The dataset $D$ contains $n = 300$ samples. For each example $\{x^{(i)}, y^{(i)}\}$, the feature dimension of the input $x^{(i)}$ is three and the corresponding label $y^{(i)} \in \{0, 1\}$ is binary. Examples of $D$ are shown in Figure 2. At the data preprocessing stage, we split the dataset $D$ into the training set $D_{tr}$, validation set $D_{va}$, and test set $D_{te}$ with size $n_{tr} = 100$, $n_{va} = 100$, and $n_{te} = 100$. The explicit form of the objective function is

$$\mathcal{L} = \frac{1}{n_{tr}} \sum_{i=1}^{n_{tr}} \left( y^{(i)}(A, x^{(i)}, \theta) - y^{(i)} \right)^2,$$

where $\{x^{(i)}, y^{(i)}\} \in D_{te}$ and $\hat{y}^{(i)}(A, x^{(i)}, \theta) \in [0, 1]$ is the output of the quantum classifier (i.e., a function taking the input $x^{(i)}$, the supernet $A$, and the trainable parameters $\theta$). The training (validation and test) accuracy is measured by $\sum_{i} \mathbb{1}_{g^{(i)} = y^{(i)}} / n_{tr}$ (sum) and $\sum_{i} \mathbb{1}_{g^{(i)} = y^{(i)}} / n_{va}$ (sum) with $g^{(i)}$ being the predicted label for $x^{(i)}$. We also apply the quantum kernel classifier proposed by [19] to learn $D$ and compare its performance.
with QAS, where the implementation of such a quantum classifier is shown in Figure 2 (b). See Appendix C for more discussion about the construction of D and the employed quantum kernel classifier.

The hyper-parameters for QAS are as follows. The number of supernets is \( W = 1 \) and \( W = 5 \), respectively. The circuit depth for all supernets is set as \( L = 3 \). The number of sampled subnets for ranking, i.e., Step 3 in Figure 1, is set as \( K = 500 \). The search space of QAS is restricted to the two qubits gates. Specifically, at each layer \( U_i(\theta) \), the qubit gates are fixed to be the rotational quantum gate along \( Y \)-axis \( R_y \). For the two qubits gates, denoted the index of three qubits as \((0, 1, 2)\). QAS explores whether applying CNOT gates to the qubits pair \((0, 1)\), \((0, 2)\), \((1, 2)\) or not. Therefore, the total number of subnets equals to \(|S| = 8^3\).

Under the noiseless scenario, the performance of QAS with three different settings is exhibited in Figure 2 (d). In particular, QAS with \( W = 1 \) and \( T = 10 \) attains the worst performance, where the validation accuracy for most supernets concentrates on \( 50\% - 60\% \), highlighted by the green bar. With increasing the number of epochs to \( T = 400 \) and fixing \( W = 1 \), the performance is slightly improved, i.e., the number of subnets that achieves validation accuracy above \( 90\% \) is 30, highlighted by the yellow bar. When \( W = 5 \) and \( T = 400 \), the performance of QAS is dramatically enhanced, where the validation accuracy of 151 subnets is above \( 90\% \). The comparison between the first two settings indicates the correctness of utilizing QAS to accomplish VQLS based learning tasks in which QAS learns useful feature information and achieves a better performance with respect to the increased epoch number \( T \). The varied performances of the last two settings reflect the fierce competition phenomenon among subnets and validate the feasibility to adopt \( W > 1 \) supernets to boost the performance of QAS. We retrain the output subnet of QAS under the setting: \( W = 5 \) and \( T = 400 \), both the training and test accuracies converge to \( 100\% \) within 15 epochs, which is identical to the original quantum kernel classifier (see Appendix C for the omitted simulation results and the exploration of fierce competition).

The performance of the original quantum kernel classifier is evidently degraded when the depolarizing error for the single qubit gate and two qubits gates are set as 0.05 and 0.2, respectively. As shown in the lower plot of Figure 2 (f), the training and test accuracies of the original quantum kernel classifier drop to 50\% (almost conduct a random guess) under the noisy setting. The degraded performance is caused by the large amount of accumulated noise, where the classical optimizer fails to receive the valid optimization information. By contrast, QAS can achieve a good performance under the same noise setting. As shown in Figure 2 (c), with setting \( W = 5 \) and \( T = 400 \), the validation accuracy of 115 subnets is above \( 90\% \) under the noisy setting. The subnet that attains the highest validation accuracy is shown in Figure 2 (c). Notably, compared with the original quantum kernel classifier in Figure 2 (b), the searched subnet contains fewer CNOT gates. This implies that, under the noisy...
The implementation of where

Under the noiseless setting, the estimated energy of VQE under the noisy and noiseless settings. The label ‘Exact’ refers to the accurate result $E_m$. (d) The performance of the output subnet of QAS under the noisy setting. (e) The performance of QAS at the ranking state. The label ‘#E = b’ refers to the number of supernets, i.e., $W = b$. The x-axis means that the estimated energy of the sampled subnet is in the range of $(c, d]$, e.g., $c = -0.6$ Ha, and $d = -0.8$ Ha.

setting formulated above, QAS suppresses the noise effect and improves the training performance by adopting few CNOT gates. When we retrain the obtained subnet with 10 epochs, both the train and test accuracies achieve 100%, as shown in the upper plot of Figure 2 (f). These results indicate the feasibility to apply QAS to achieve the error mitigation and trainability enhancement.

We next apply QAS to find the ground state energy of the Hydrogen molecule [12] [13] under both the noisless and noisy scenarios. The molecular hydrogen Hamiltonian is formulated as

$$H_h = g + \sum_{i=0}^{3} g_i Z_i +$$

$$+ \sum_{i=1,k=1,i<k}^{3} g_{i,k} Z_i Z_k + g_a Y_0 X_1 X_2 Y_3$$

$$+ g_b Y_0 Y_1 X_2 X_3 + g_c X_0 X_1 Y_2 Y_3 + g_d X_0 Y_1 Y_2 X_3,$$

where $\{X_i, Y_i, Z_i\}$ denote the Pauli matrices acting on the $i$-th qubit and the real scalars $g$ with or without subscripts are efficiently computable functions of the hydrogen-hydrogen bond length (see Appendix D for details about $H_h$ and $g$). The ground state energy calculation amounts to compute the lowest energy eigenvalues of $H_h$, where the accurate value is $E_m = -1.136$ Ha [39].

To tackle this task, the conventional variational quantum Eigen-solver (VQE) [14] and its variants [14] [40] optimize the trainable parameters in $U(\theta)$ to prepare the ground state $|\psi^*\rangle = U(\theta^*) |0\rangle \otimes |0\rangle$ of $H_h$, i.e., $E_m = \langle \psi^* | H | \psi^* \rangle$. The implementation of $U(\theta)$ is illustrated in Figure 3 (a). Under the noiseless setting, the estimated energy of VQE fast converges to the target result $E_m$ within 40 iterations, as shown in Figure 3 (c).

The hyper-parameters of QAS to compute the lowest energy eigenvalues of $H_h$ are as follows. The number of supernets has two settings, i.e., $W = 1$ and $W = 5$, respectively. The circuit depth for all supernets is $L = 3$. The number of iterations and sampled subnets for ranking is $T = 500$ and $K = 500$, respectively. The search space of QAS for the single qubit gates is fixed to be the rotational quantum gates along $Y$ and $Z$-axis, i.e., $R_y$ and $R_z$. For the two qubits gates, denoted the index of four qubits as $(0, 1, 2, 3)$, QAS explores whether applying CNOT gates to the qubits pair $(0, 1), (1, 2), (2, 3)$ or not. Therefore, the total number of subnets equals to $|S| = 128^3$. The performance of QAS with $W = 5$ is shown in Figure 3 (d). Through retraining the obtained subnet of QAS with 50 iterations, the estimated energy converges to $E_m$, which is the same with the conventional VQE.

The performance between the conventional VQE and QAS is largely distinct when the noisy model described in the classification task is deployed. Due to the large amount of gate noise, the estimated ground energy of the conventional VQE converges to $-0.4$ Ha, as shown in Figure 3 (e). In contrast, the estimated ground energy of QAE with $W = 1$ and $W = 5$ achieves $-0.93$ Ha and $-1.05$ Ha, respectively. Both of them are closer to the target result $E_m$ compared with the conventional VQE. Moreover, as shown in Figure 3 (e), more supernets imply a better performance of QAE, since the estimated energy
of most subnets is below $-0.6\text{Ha}$ when $W = 5$, while the estimated energy of 350 subnets is above 0Ha when $W = 1$. We illustrate the generated subnet of QAS with $W = 5$ in Figure 3 (b). In particular, to mitigate the effect of gate noise, this generated subnet does not contain any CNOT gate, which is applied to a very large noise level. Recall that a central challenge in quantum computational chemistry is whether NISQ devices can outperform classical methods already available [27]. The achieved results in QAS can provide a good guidance to answer this issue. Concretely, the searched subnet in Figure 3, which only produces the separable states that can be efficiently simulated by classical devices, suggests that VQE method may not outperform classical methods when NISQ devices contain large gate noise.

Note that more simulation results of the above task are given in Appendix D. Furthermore, we implement the conventional VQE and QAS on the real superconducting quantum hardware, i.e., ‘ibmq_ourense’, to estimate the ground state energy of $H_b$. Experimental result indicates that the subnet obtained by QAS outperforms the conventional VQE, where the estimated energy of the former is $-0.96\text{Ha}$ while the latter is $-0.61\text{Ha}$.

We last exhibit how QAS contributes to the alleviation of the barren plateau issue [21]. Recall one conclusion of the barren plateau is that the gradient vanishes exponentially in the number of circuit layers $L$. Mathematically, the averaged gradient norm of the objective function in Eqn. (1) will fast converge to zero with respect to $L$, i.e., $\|\nabla_\theta \mathcal{L}(\theta)\|/d \sim O(e^{-L})$. To exhibit that QAS can alleviate the barren plateau phenomenon, we evaluate the gradient norm of VQE and QAS when they are applied to accomplish the task of finding ground state energy of $H_b$ in Eqn. (7). In particular, we iteratively increase the circuit depth of VQE and QAS from $L = 2$ to $L = 7$, and calculate the variance of the gradient norm $\|\nabla_\theta \mathcal{L}(\theta)\|/d$ with respect to the observable $Z_0$ in $H_b$, where $\theta$ are randomly sampled from the uniform distribution with 2000 times. The simulation results under the noiseless setting are shown in Figure 4. As for the conventional VQE, the variance of the gradient norm is continuously decreased when $L$ is increased, which accords with the theoretical results. By contrast, for QAS, the variance of the gradient norm with $L = 8$ is even larger than that of $L = 4$. This result implies that QAS can alleviate the influence of the barren plateau for large $L$. Through checking the output subnet, we observe that QAS tends to decrease the usage of CNOT gates in order to relieve barren plateaus.

We remark that QAS, as a general scheme, can efficiently integrate with other advanced machine learning techniques to further improve its performance. Concretely, in Appendix E, we elaborate on how to use evolutionary algorithms to advance the ranking stage of QAS.

IV. Discussion

In this study, we devise QAS to achieve the error mitigation and trainability enhancement for VQLS based learning algorithms. Both simulation and experimental results validate the efficiency of QAS. Besides the good performance, QAS is resource and runtime efficient, and is compatible with all quantum systems. To the best of our knowledge, this is the first proposal that can be directly applied to current quantum machines to achieve the error mitigation and the trainability enhancement for VQLS with scalability. In addition, through incorporating QAS with other advanced error mitigation techniques and barren plateau alleviation approaches, it is possible to seek more applications that can be implemented on NISQ machines with potential advantages.

There are many natural questions remaining in the study of QAS. Our future work includes the following three directions. First, we will explore better strategies to sample the subnet at each iteration and more effective methods to allocate the subnet into different supernets. For example, the reinforcement learning techniques, which is used to construct optimal sequences of unitaries to accomplish quantum simulation tasks [28], may contribute to this goal. Second, we will design a more advanced weight sharing strategy to reduce the parameters space of the supernet from $O(LQ^{2k})$ to $O(LQN)$. Last, we aim to leverage some prior information of noisy model to boost the performance of QAS.

Note added:

During the preparation of the manuscript, we notice a relevant paper proposed by Zhang, Hsieh, Zhang, and Hong Yao (arXiv:2010.08561), which could improve the performance of several tasks related to unitary decomposition and quantum approximate optimization algorithms. In that work the authors also employed a strategy to automate quantum circuit architectures. Their proposal can be treated as a classical generalization of the differentiable neural architecture search (DARTS), while the building block of our proposal is the one-shot neural architecture search (NAS) [29]. Considering that DARTS
and one-shot NAS are two main branches in the field of auto-machine learning [49]; our work together with Zhang et al., as their quantum generalizations, contribute to a wide range of quantum learning tasks involving circuit optimization. Finally, beyond extending classical results to the quantum scenario, we identify the difference between NAS and QAS i.e., the fierce competition phenomenon, and propose an effective approach to tackle this issue.

[1] Daniel A Lidar and Todd A Brun. Quantum error correction. Cambridge university press, 2013.
[2] John Preskill. Quantum computing in the nisq era and beyond. Quantum, 2:79, 2018.
[3] David G Cory, MD Price, W Maas, Emanuel Knill, Raymond Laflamme, Wojciech H Zurek, Timothy F Havel, and Shyamal S Somaroo. Experimental quantum error correction. Physical Review Letters, 81(10):2152, 1998.
[4] Todd Brun, Igor Devetak, and Min-Hsiu Hsieh. Correcting quantum errors with entanglement. Science, 314(5798):436–439, 2006.
[5] Nissim Ofek, Andrei Petrenko, Reinier Heeres, Philip Reinhold, Zaki Leghtas, Brian Vlastakis, Yehan Liu, Luigi Frunzio, SM Girvin, Liang Jiang, et al. Extending the lifetime of a quantum bit with error correction in superconducting circuits. Nature, 536(7617):441–445, 2016.
[6] Christian Kraglund Andersen, Ants Remm, Stefania Lazar, Sebastian Krinner, Nathan Lacroix, Graham J Norris, Mihai Gabureac, Christopher Eichler, and Andreas Wallraff. Repeated quantum error detection in a surface code. Nature Physics, pages 1–6, 2020.
[7] Earl T Campbell, Barbara M Terhal, and Christophe Vuillot. Roads towards fault-tolerant universal quantum computation. Nature, 549(7671):172–179, 2017.
[8] Kristan Temme, Sergey Bravyi, and Jay M Gambetta. Error mitigation for short-depth quantum circuits. Physical review letters, 119(18):180509, 2017.
[9] Suguru Endo, Simon C Benjamin, and Ying Li. Practical quantum error mitigation for near-future applications. Physical Review X, 8(3):031027, 2018.
[10] Ying Li and Simon C Benjamin. Efficient variational quantum simulator incorporating active error minimization. Physical Review X, 7(2):021050, 2017.
[11] Jarrod R McClean, Mollie E Kimchi-Schwartz, Jonathan Carter, and Wibe A De Jong. Hybrid quantum-classical hierarchy for mitigation of decoherence and determination of excited states. Physical Review A, 95(4):042308, 2017.
[12] Armands Strikis, Dayue Qin, Yanzhu Chen, Simon C Benjamin, and Ying Li. Learning-based quantum error mitigation. arXiv preprint arXiv:2005.07601, 2020.
[13] Piotr Czarnik, Andrew Arrasmith, Patrick J Coles, and Lukasz Cincio. Error mitigation with Clifford quantum-circuit data. arXiv preprint arXiv:2005.10189, 2020.
[14] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O brien. A variational eigenvalue solver on a photonic quantum processor. Nature communications, 5:4213, 2014.
[15] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. arXiv preprint arXiv:1802.06002, 2018. URL https://arxiv.org/abs/1802.06002
[16] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised learning with quantum-enhanced feature spaces. Nature, 567(7747):209, 2019.
[17] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. Physical review letters, 122(4):040504, 2019.
[18] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized quantum circuits as machine learning models. Quantum Science and Technology, 4(4):043001, 2019.
[19] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. Expressive power of parametrized quantum circuits. Phys. Rev. Research, 2:033125, Jul 2020. doi:10.1103/PhysRevResearch.2.033125 URL https://link.aps.org/doi/10.1103/PhysRevResearch.2.033125
[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016.
[21] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. Nature communications, 9(1):1–6, 2018.
[22] M Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles. Cost-function-dependent barren plateaus in shallow quantum neural networks. arXiv preprint arXiv:2001.00550, 2020.
[23] Kevin J. Sung, Matthew P. Harrigan, Nicholas C. Rubin, Zhang Jiang, Ryan Babbush, and Jarrod R. McClean. An exploration of practical optimizers for variational quantum algorithms on superconducting qubit processors, 2020.
[24] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Dacheng Tao, and Nana Liu. Quantum noise protects quantum classifiers against adversaries. arXiv preprint arXiv:2003.09416, 2020.
[25] Samson Wang, Enrico Fontana, Marco Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J. Coles. Noise-induced barren plateaus in variational quantum algorithms. arXiv preprint arXiv:2007.14384, 2020.
[26] Google AI Quantum et al. Hartree-fock on a superconducting qubit quantum computer. Science, 369(6507):1084–1089, 2020.
[27] D Chivilikhin, A Samarain, V Ulyantsev, I Iorsh, AR Oganov, and O Kyriienko. Mog-vqe: Multiobjective genetic variational quantum eigensolver. arXiv preprint arXiv:2007.04424, 2020.
[28] Li Li, Minjie Fan, Marc Coram, Patrick Riley, Stefán Leichenauer, et al. Quantum optimization with a novel gibbs objective function and ansatz architecture search. Physical Review Research, 2(2):023074, 2020.
[29] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. Journal of Machine Learning Research, 20:1–21, 2019.
[30] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In International Conference on Machine Learning, pages 4095–4104, 2018.
[31] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille,
Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In Proceedings of the European Conference on Computer Vision (ECCV), pages 19–34, 2018.

[32] Shan You, Tao Huang, Mingmin Yang, Fei Wang, Chen Qian, and Changshui Zhang. Greedynas: Towards fast one-shot nas with greedy supernet. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1999–2008, 2020.

[33] Yibo Yang, Hongyang Li, Shan You, Fei Wang, Chen Qian, and Zhouchen Lin. Ista-nas: Efficient and consistent neural architecture search by sparse coding. arXiv preprint arXiv:2010.06176, 2020.

[34] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. arXiv preprint arXiv:1204.5721, 2012.

[35] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. Implementable quantum classifier for nonlinear data. arXiv preprint arXiv:1809.06056, 2018.

[36] Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. Nature Physics, 15(12):1273–1278, 2019.

[37] Ryan LaRose, Arkin Tikku, Étude O’Neel-Judy, Lukasz Cincio, and Patrick J Coles. Variational quantum state diagonalization. npj Quantum Information, 5(1):1–10, 2019.

[38] Yuxuan Du, Min-Hsiu Hsieh, and Dacheng Tao. Efficient online quantum generative adversarial learning algorithms with applications. arXiv preprint arXiv:1904.09602, 2019.

[39] Ville Bergholm, Josh Izacq, Maria Schulz, Christian Gogolin, Carsten Blank, Keri McKiernan, and Nathan Killoran. Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv preprint arXiv:1811.04968, 2018.

[40] Qiskit: An open-source framework for quantum computing, 2019.

[41] https://github.com/yuxuan-du/Quantum_architecture_search/

[42] Peter JJ O’Malley, Ryan Babbush, Ian D Kivlichan, Jonathan Romero, Jarrod R McClean, Rami Barends, Julian Kelly, Pedram Roushan, Andrew Tranter, Nan Ding, et al. Scalable quantum simulation of molecular energies. Physical Review X, 6(3):031007, 2016.

[43] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature, 549(7671):242–246, 2017.

[44] Daochen Wang, Oscar Higgott, and Stephen Brierley. Accelerated variational quantum eigensolver. Physical review letters, 122(14):140504, 2019.

[45] James Stokes, Josh Izaac, Nathan Killoran, and Giuseppe Carleo. Quantum natural gradient. Quantum, 4:269, 2020.

[46] Kosuke Mitarai, Tennyin Yan, and Keisuke Fujii. Generalization of the output of a variational quantum eigensolver by parameter interpolation with a low-depth ansatz. Physical Review Applied, 11(4):044087, 2019.

[47] Sam McArdle, Suguru Endo, Alan Aspuru-Guzik, Simon C Benjamin, and Xiao Yuan. Quantum computational chemistry. Reviews of Modern Physics, 92(1):015003, 2020.

[48] Jiabao Yao, Lin Lin, and Marin Bukov. Reinforcement learning for many-body ground state preparation based on counter-diabatic driving. arXiv preprint arXiv:2010.03655, 2020.

[49] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In Computer Vision – ECCV 2020, pages 544–560, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58517-4.

[50] Sébastien Gerchinovitz and Tor Lattimore. Refined lower bounds for adversarial bandits. In Advances in Neural Information Processing Systems, pages 1198–1206, 2016.

[51] W Pirie. Spearman rank correlation coefficient. Encyclopedia of statistical sciences, 12, 2004.

[52] M. G. Kendall. A new measure of rank correlation. Biometrika, 30(1/2):81–93, 1938.

[53] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE transactions on evolutionary computation, 6(2):182–197, 2002.

[54] Richard S Sutton et al. Introduction to reinforcement learning, volume 135.
We organize the appendix as follows. In Appendix A, we elaborate on the implementation of the supernet and the weight sharing strategy. In Appendix B, we establish the connection between the bandit learning and the subnet assignment task, provide the proof of Theorem 1 and discuss how to exploit bandit learning algorithms to further advance the subnet assignment task. We then provide explanations and simulation results related to the fierce competition phenomenon and the classification task in Appendix C. Afterwards, we present simulation and experiment details about the quantum chemistry tasks in Appendix D. Last, we exhibit how to introduce evolutionary algorithms into the ranking stage to boost the performance of QAS in Appendix E.

A. Supernet and the weight sharing strategy

**Supernet and Subnets.** Supernet is an abstract concept inherited from the neural network architecture search [29]. The aim of the supernet in QAS is manipulating all possible quantum circuit architectures of \( U(\theta) \), training them efficiently, and outputting the candidate with the best performance. Recall the objective function in Eqn. (3). The output circuit architecture is the one that can achieve the minimum objective value among all circuit architectures when they are training independently.

![Diagram of Supernet and Subnets](image)

**FIG. 5: The weight sharing strategy.** The red box for all three plots refers to the sequence of the sampled quantum gates in \( U_l(\theta) \), while the black boxes represent other trainable unitary layers \( \{U_j(\theta)\} \) with \( j \neq l \). The \( U_i(\theta) \) in the left and middle circuits, highlighted by the green region, share the same training parameters. The \( U_i(\theta) \) in the right circuit, highlighted by the red region, does not share the parameters with the other two quantum circuits.

For ease of description, we restrict that the allowed quantum operations in each layer \( U_l(\theta) \) are as follows. First, each single qubit is only operated with one single-qubit gate sampled from \( \{R_X, R_Y, R_Z\} \). Second, for any pair of qubits, we can apply a CNOT gate to them or do nothing. In addition, let the number of qubits be \( N \) and the maximum layer number of \( U(\theta) \) in Eqn. (2) be \( L \). Under such a setting, the total number of gates arrangements to construct \( U_l(\theta) \) is \( 3^N2^{C(2^N)} \), where the terms \( 3^N \) and \( 2^{C(2^N)} \) refer to all possible arrangements for single qubit gates and CNOT gates, respectively. Since there are in total \( L \) layers, the number of gates arrangements to construct \( U(\theta) \) is \( (3^N2^{C(2^N)})^L \). The supernet used in QAS amounts to a manager that controls and optimizes these \( (3^N2^{C(2^N)})^L \) circuit architectures. A subnet corresponds to a particular gates arrangement to fabricate \( U(\theta) \). The goal of QAS is that after training \( T \) iterations, i.e., \( T \ll (3^N2^{C(2^N)})^L \), the supernet can output a subnet as the approximated solution of Eqn. (2).

**Weight sharing strategy.** As explained above, the number of gates arrangements in the supernet of QAS exponentially scales with \( N \) and \( L \). When we initialize the trainable parameters for each possible circuit architectures independently, the classical memory needs to store \( 3^{NL} \) parameters, which becomes unaffordable for large \( N \) and \( L \). Moreover, at each training step, the achieved optimization information for the sampled subnet is useless for the rest subnets. In other words, it is intractable to train all subnets when \( T \ll (3^N2^{C(2^N)})^L \).

To relieve the above two issues, the weight sharing strategy is exploited to avoid training each subnet from scratch, and to force all subnets to share trainable parameters. In doing so, the time (i.e., the number of iterations \( T \) to seek the optimal subnet from exponential subnets is dramatically reduced. The weight sharing strategy adopted in QAS can be formulated as follows. For all subnets, once the sequence of single qubit gates in \( U_l(\theta) \) is identical, they share the same trainable parameters in this layer. A concrete example is shown in Figure 5. In particular, the sequence of single qubit gates of \( U_l(\theta) \) for the left and middle circuits (subnets) is identical, which is in the order of \( [R_Y(\theta_{1,1}), R_Z(\theta_{1,2}), R_Y(\theta_{1,3}), R_Y(\theta_{1,4})] \). In other words, for all subnets whose arrangement of single qubit gates in the \( l \)-th layer \( U_l(\theta) \) are same, the trainable parameters \( \{\theta_{l,i}\}_{i=1}^4 \) are sharing, no matter how the connection of CNOT gates is varied. A counterexample is shown in the middle and right panels of Figure 5. Although their CNOT gates arrangements in \( U_l(\theta) \) are same, these two circuits cannot share the parameters \( \{\theta_{l,i}\}_{i=1}^4 \) in the \( l \)-th layer, since the sequence of single qubit gates is different.

The weight sharing strategy enables QAS to use \( (3^N2^{C(2^N)})L \) trainable parameters to optimize \( (3^N2^{C(2^N)})^L \) subnets. The linearly dependence on \( L \) is crucial to alleviate the burden of classical memories to store the trainable parameters.
Furthermore, the weight sharing strategy allows the acceleration of the optimization, i.e., at each training step, the optimizing information of the sampled subnet can be reused to update other subnets that shares the same parameters.

We remark that it is intriguing to design other weight sharing strategies to further improve the training and memory efficiencies of QAS.

**W supernets.** To better improve the learning performance and to resolve the fierce competition among different subnets, QAS employs \( W > 1 \) supernets in the training and ranking stages. Notably, the mechanism of \( W \) supernets is exactly equivalent to the single supernet case, i.e., the weight sharing strategy is applied to each supernet. In the training process, the parameters of \( W \) supernets are independently optimized, which implies that the total number of parameters contained in QAS is \( (3^{N}2^{C^2N})LW \). At each iteration, the sampled subnet is separately evaluated by these \( W \) supernets. After evaluation, the supernet, whose objective value is minimal among all \( W \) results, can update the corresponding parameters. Meanwhile, the trainable parameters of the rest supernets remain unchanged.

**B. The subnet assignment task**

In this section, we first connect the subnet assignment task with the adversarial bandit learning problem. We then provide the proof of Theorem 1. We last explain how to employ advanced bandit learning algorithms to reduce the runtime complexity of the subnet assignment task.

1. **The connection between the adversarial bandit learning and the subnet assignment**

Let us first introduce the adversarial bandit learning. In the adversarial bandit learning [33], a player has \( W \) possible arms to choose. Denote the total number of iterations as \( T \). At the \( t \)-th iteration,

- The player chooses an arm \( w^{(t)} \in [W] \) with a deterministic strategy or sampling from a certain distribution \( P_w \);
- The adversary chooses a cost \( c^{(t)}(w^{(t)}) \) for the chosen arm \( w^{(t)} \);
- The cost of the selected arm \( w^{(t)} \), i.e., \( c^{(t)}(w^{(t)}) \) with \( w^{(t)} \in [W] \), is revealed to the player.

The goal of the adversarial bandit learning is minimizing the total cost over \( T \) iterations, where its performance is quantified by the regret \( r_T \), i.e.,

\[
    r_T = \sum_{t=1}^{T} c^{(t)}(w^{(t)}) - \min_{w \in [W]} \sum_{t=1}^{T} c^{(t)}(w).
\]

Intuitively, the regret \( r_T \) compares the cumulative cost of the selected arms \( \{w^{(t)}\}_{t=1}^{T} \) with the best arm in hindsight. If \( r_T = o(T) \), where the regret can be either negative or scales at most sublinearly with \( T \), we say that the player is learning; otherwise, when \( r_T = \Theta(T) \) such that the regret scales linearly with \( T \), we say that the player is not learning, since the averaged cost per-iteration does not decrease with time.

We now utilize the language of the adversarial bandit learning to restate the subnet assignment problem. In QAS, each arm refers to a supernet and the number of arms equals to the number of supernets. The cost \( c^{(t)}(w^{(t)}) \) is equivalent to the objective function \( \mathcal{L}(\theta^{(t,w)}, a^{(t)}) \) in Eqn. (3), where \( a^{(t)} \) refers to the sampled subnet \( a^{(t)} \in \mathcal{S} \), and \( \theta^{(t,w)} \) represents the trainable parameters of the \( w \)-th supernet \( A^{(w)} \). The aim of the subnet assignment is to allocate \( \{a^{(t)}\}_{t=1}^{T} \) to the best sequence of arms (supernets) to minimize the cumulative cost. Denote the selected sequence of arms (indices of supernets) of QAS as \( \{i^{(t)}_{w}\}_{t=1}^{T} \). The regret in Eqn. (B1) can be rewritten as

\[
    \hat{R}_T = \sum_{t=1}^{T} \mathcal{L}(\theta^{(i^{(t)}_{w}(t)), a^{(t)}}) - \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)}).
\]

2. **Proof of Theorem 1**

The proof of Theorem 1 exploits the following lemma.
Lemma 1 (Theorem 1, [50]). Suppose \( W \geq 2 \) and \( \delta \in (0, 1/4) \) and \( T \geq 32(W - 1)\log(2/\delta) \), then there exists a sequence of data \( \{\theta^{(t)}, I^{(t)}\}^{T}_{t=1} \), or equivalently, the objective values \( \{\mathcal{L}(\theta^{(t)}, a^{(t)})\}^{T}_{t=1} \) such that the regret in Eqn. (B2) follows

\[
P \left( \tilde{R}_T \geq \frac{1}{2\tilde{c}} \sqrt{(W - 1)T\log(1/(4\delta))} \right) \geq \delta/2. \tag{B3}
\]

The lower bound given in Lemma 1 indicates that under the adversarial setting, there does not exist an adversarial bandit algorithm that can achieve the regret smaller than \( \Omega(\sqrt{WT\log(1/\delta)}) \) with probability at least \( 1 - \delta \).

We are now ready to prove Theorem 1.

Proof of Theorem 1. Here we first prove the regret \( R_T \) in Eqn. (3) for the assignment strategy employed in QAS. We then quantify the lower bound of \( \tilde{R}_T \) for all adversarial bandit algorithms.

Recall the assignment strategy used in QAS. Given the sampled subnet \( a^{(t)} \in S \), QAS feeds this subnet into \( W \) supernets and compares \( W \) values of objective functions, i.e., \( \{\mathcal{L}(\theta^{(t,w)}, a^{(t)})\}^{W}_{w=1} \). Then, the subnet \( a^{(t)} \) is assigned to the \( I^{(t)}_w \)-th supernet as

\[
I^{(t)}_w = \arg\min_{w=1,...,W} \mathcal{L}(\theta^{(t,w)}, a^{(t)}). \tag{B4}
\]

By exploiting the explicit definition of \( I^{(t)}_w \) in Eqn. (B4), the regret \( R_T \) in Eqn. (5) yields

\[
R_T = \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,I^{(t)}_w)}, a^{(t)}) - \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)})
\]

\[
= \sum_{t=1}^{T} \min_{w=1,...,W} \mathcal{L}(\theta^{(t,w)}, a^{(t)}) - \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)})
\]

\[
\leq 0, \tag{B5}
\]

where the last inequality employs the fact that the summation of minimum values of functions is less than the minimum value of summation of functions (i.e., \( \sum_{t=1}^{T} \min_{w \in [W]} \mathcal{L}(\theta^{(t,w)}, a^{(t)}) \) and the equality is hold when the minimum of all functions \( \{f_i(x)\} \) is identical).

We next prove that the regret of all bandit algorithms are lower bounded by \( \Omega(\sqrt{WT}) \). Specifically, the discrepancy between the regret \( R_T \) in Eqn. (5) and \( \tilde{R}_T \) in Eqn. (B2) yields

\[
R_T - \tilde{R}_T = \left( \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,I^{(t)}_w)}, a^{(t)}) - \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)}) \right) - \left( \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,I^{(t)}_w)}, a^{(t)}) - \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)}) \right)
\]

\[
= \sum_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)}) - \sum_{t=1}^{T} \min_{w \in [W]} \sum_{t=1}^{T} \mathcal{L}(\theta^{(t,w)}, a^{(t)})
\]

\[
\geq 0. \tag{B6}
\]

In conjunction with Eqn. (B6) and Lemma 1 we achieve

\[
P \left( R_T \geq \frac{1}{2\tilde{c}} \sqrt{(W - 1)T\log(1/(4\delta))} \right) \geq \delta/2. \tag{B7}
\]

In other words, for the subnet assignment task, there does not exist an adversarial bandit algorithm that can achieve \( R_T \leq \Omega(\sqrt{WT\log(1/\delta)}) \) with probability at least \( 1 - \delta \).

Based on Eqn. (B5) and Eqn. (B7), we conclude that no bandit learning algorithm can achieve a lower regret than that of the strategy adopted in QAS.

3. Applying bandit learning algorithms to the subnet assignment task

Here we discuss how to apply bandit learning algorithms to improve the subnet assignment task in terms of the runtime cost. Recall the subnet assignment strategy used in QAS. At each iteration, the sampled subnet should feed into \( W \) supernets separately and then compare the returned \( W \) objective values. In this way, the runtime complexity becomes expensive for a large \( W \). The adversarial bandit learning algorithms are a promising solution to tackle the runtime issue. As explained in Appendix B1, when adversarial bandit learning algorithms are employed, the subnet is only required to feed into one supernet at each iteration, while the price is inducing a relatively large regret bound.
C. The synthetic dataset classification task

The outline of this section is as follows. In Appendix [C1], we first introduce some basic terminologies in machine learning to make our description self-consistent. In Appendix [C2], we explain how to construct the synthetic dataset \( \mathcal{D} \). In Appendix [C3], we provide the simulation results omitted in the main text and elaborate on the fierce competition phenomenon.

1. Basic terminologies in machine learning

When we apply QAS to accomplish the classification task, the terminology ‘epoch’, which is broadly used in the field of machine learning [20], is employed to replace ‘iteration’. Intuitively, an epoch means that an entire dataset is passed forward through the quantum learning model. For the quantum kernel classifier used in the main text, each training example in \( \mathcal{D}_{tr} \) is fed into the quantum circuit in sequence to acquire the predicted label. Since \( \mathcal{D}_{tr} \) includes in total 100 examples, it will take 100 iterations to complete one epoch.

In the synthetic classification task, we split the datasets into three parts, i.e., the training, validation, and test datasets, following the convention of machine learning [20]. The training dataset \( \mathcal{D}_{tr} \) is used to optimize the trainable parameters during the learning process. The function of the validation dataset \( \mathcal{D}_{va} \) is estimating how well the classifier has been trained. During \( T \) epochs, the trainable parameters that achieve the highest validation accuracy are set as the output parameters. Mathematically, the output parameters satisfy

\[
\hat{\theta} = \max_{\{\theta^{(l)}\}_{l=1}^T} \sum_{i} \mathbb{1}_{\hat{y}^{(i)}(\theta^{(l)}, x^{(i)}) = y^{(i)}},
\]

where \( \{x^{(i)}, y^{(i)}\} \in \mathcal{D}_{va} \), \( \hat{y}^{(i)} \) is the prediction of the classifier given \( \theta^{(l)} \) and \( x^{(i)} \), and \( \mathbb{1}_{z} \) is the indicator function that takes the value 1 if the condition \( z \) is satisfied and zero otherwise. Finally, the output parameters \( \hat{\theta} \) are applied to the test dataset to benchmark the performance of the trained classifier.

2. Implementation of the synthetic dataset

Here we recap the method to construct the synthetic dataset proposed in [16]. Denote the encoding layer as

\[
U_x = R_Y(x_1) \otimes R_Y(x_2) \otimes R_Y(x_3).
\]

To establish the synthetic dataset \( \mathcal{D} \) used in the main text, we first generate a set of data points \( \{x^{(i)}\} \) with \( x^{(i)} \in \mathbb{R}^3 \). We then define the optimal circuit as

\[
U^*(\theta^*) = \prod_{l=1}^{3} U^*_l(\theta^*_l),
\]

where \( U^*_l(\theta^*_l) = \otimes_{j=1}^{3} R_Y(\theta^*_l,j)(\text{CNOT} \otimes I_2)(I_2 \otimes \text{CNOT}) \) and the parameter \( \theta^*_l,j \) is uniformly sampled from \([0, 2\pi)\) for all \( j \in [3] \) and \( l \in [3] \). The strategy to label \( x^{(i)} \) is as follows. Let \( \Pi = I_4 \otimes |0\rangle \langle 0| \) be the measurement operator. The data point \( x^{(i)} \) is labeled as \( y^{(i)} = 1 \) if

\[
\langle 000|U_{x^{(i)}}^\dagger U^*(\theta^*)\Pi U^*(\theta^*) U_{x^{(i)}}|000\rangle \geq 0.75.
\]

The label of \( x^{(i)} \) is assigned as \( y^{(i)} = 0 \) if

\[
\langle 000|U_{x^{(i)}}^\dagger U^*(\theta^*)\Pi U^*(\theta^*) U_{x^{(i)}}|000\rangle \leq 0.25.
\]

Note that, if the measured result is in the range \((0.25, 0.75)\), we drop this data point and sample a new one. By repeating the above procedure, we can built the synthetic dataset \( \mathcal{D} \).

3. Simulation results of the synthetic dataset classification and the fierce competition phenomenon

Here we first introduce how to use the quantum kernel classifier to conduct the prediction. Given the data point \( x^{(i)} \in \mathcal{D} \) at the \( t \)-th epoch, the quantum kernel classifier is composed of two unitraies, i.e., \( U_{x^{(i)}} \) and \( U(\theta^{(l)}) \), where the
sequence of quantum gates in \( U(\theta^{(t)}) \) is fixed as shown in Figure 2 (b). The output of quantum kernel classifier yields
\[
\tilde{y}(x^{(i)}, \theta^{(t)}) = \langle 000 | U_{x^{(i)}}^\dagger U(\theta^{(t)}) \Pi U(\theta^{(t)}) U_{x^{(i)}} | 000 \rangle.
\]  
(C6)

The predicted label of \( x^{(i)} \), i.e., \( g(\tilde{y}(x^{(i)}, \theta^{(t)})) \), becomes
\[
g(\tilde{y}(x^{(i)}, \theta^{(t)})) = \begin{cases} 
0, & \text{if } \tilde{y}(x^{(i)}, \theta^{(t)}) < 0.5 \\
1, & \text{otherwise}
\end{cases}.
\]  
(C7)

When QAS is employed to enhance the trainability and to mitigate error of the quantum kernel classifier, the arrangement of quantum gates in \( U(\theta) \) is no longer fixed and depends on the sampled subnet. In other words, at the \( t \)-th epoch, given the data point \( x^{(i)} \in \mathcal{D} \), the measured result \( \tilde{y}(A, x^{(i)}, \theta^{(t)}) \) in Eqn. (6) is
\[
\tilde{y}(A, x^{(i)}, \theta^{(t)}) = \langle 000 | U_{x^{(i)}}^\dagger U(\theta^{(t)}) a \Pi U(\theta^{(t)}) U_{x^{(i)}} a | 000 \rangle,
\]  
(C8)

where \( U(\theta^{(t)}, a) \) denotes that the trainable unitary amounts to the subnet \( a \) and the corresponding trainable parameters \( \theta^{(t)} \) are controlled by the supernet \( A \).

We then provide the simulation results of the conventional quantum kernel classifier and QAS towards the synthetic dataset \( \mathcal{D} \) under the noiseless setting. As exhibited in Figure 6 (a), both the training and validation accuracies of the conventional quantum kernel classifier fast converge to 100% after 80 epochs. The test accuracy also reaches 100%, highlighted by the green marker. Meanwhile, the loss \( \mathcal{L} \) decreases to 0.24. These results indicate that the conventional quantum kernel classifier with the protocol as depicted in Figure 2 (b) can well learn the synthetic dataset \( \mathcal{D} \).

The hyper-parameters of QAS under the noiseless setting are identical to the noisy setting introduced in the main text. Specifically, we set \( T = 400 \) and \( W = 1 \) in the training stage (Step 2), \( K = 500 \) in the ranking stage (Step 3), and \( T = 10 \) in the retraining stage (Step 4). Figure 6 (b) demonstrates the output subnet in Step 3. Compared to the conventional quantum kernel classifier, the output subnet includes fewer CNOT gates, which is more amiable for physical implementations. Figure 6 (c) illustrates the learning performance of the output subnet in the retraining stage. Concretely, both the training and test accuracies converge to 100% after one epoch. These results indicate that QAS can well learn the synthetic dataset \( \mathcal{D} \) under the noiseless setting. Note that for all simulation results related to classification tasks, the Adam optimizer [20] is exploited to update the training parameters of the quantum kernel classifier and QAS. The learning rate is set as 0.05.

**FIG. 6: Simulation results for the synthetic data classification.** (a) The performance of the conventional quantum classifier under the noiseless setting. The label ‘Train_acc’, ‘Valid_acc’, and ‘Test_acc’ refers to the training, validation, and test accuracy, respectively. (b) The simulation results of QAS in the retraining stage under the noiseless setting. (c) The subnet (i.e., the quantum circuit architecture) generated by QAS in Step 3.

We end this subsection by explaining the fierce competition phenomenon encountered in the optimization of QAS. Namely, when the number of supernets is 1, some subnets that can achieve high classification accuracies with independently training, will perform poorly in QAS. To exhibit that QAS indeed searches a set of subnets (quantum circuit architectures) with high classification accuracies, we examine the correlation of the performance of the subnet with independently optimization and training by QAS. In particular, we randomly sample 500 subnets from all possible architectures and evaluate the widely-used Spearman and Kendall tau rank correlation coefficients [51, 52], which are in the range of \([0, 1]\). In particular, larger correlation coefficients (or equivalently, stronger correlations) indicate that the ranking distribution achieved by QAS is consistent with the performance of different circuit architectures with independently training. Moreover, larger correlation coefficients also imply that the output subnet of QAS can well estimate the target subnet \( a^* \) in Eqn. (3).

The Spearman rank correlation coefficient \( \rho_S \) quantifies the monotonic relationships between random variables \( r \) and \( s \). Specifically, the spearman rank correlation coefficient between \( r \) and \( s \) is defined as
\[
\rho_S = \frac{cov(r, s)}{\sigma_r \sigma_s},
\]  
(C9)
where \( \text{cov}(\cdot, \cdot) \) is the covariance of two variables, and \( \sigma_r (\sigma_s) \) refers to the standard deviations of \( r (s) \). Suppose that \( r \in \mathbb{R}^n \) and \( s \in \mathbb{R}^n \) are two observation vectors of \( r \) and \( s \), respectively, the explicit form \( \rho_S \) is

\[
\rho_S = 1 - \frac{6 \sum_{i=1}^{n} (r_i - s_i)^2}{n(n^2 - 1)}.
\] (C10)

When the Spearman rank correlation is employed in QAS, the observation vector \( r \) (s) corresponds to the achieved validation accuracy of the sampled 500 subnets in the ranking stage, while the observation vector \( s \) corresponds to the achieved validation accuracy of the sampled 500 subnets with independently training.

The Kendall tau rank correlation coefficient concerns the relative difference of concordant pairs and discordant pairs. Specifically, in QAS, denote \( r \) (s) as the observation vector that refers to the achieved validation accuracy of the sampled 500 subnets in the ranking stage (with independently training). Given any pair \((r_i, r_j)\) and \((s_i, s_j)\), it is said to be concordant if \((r_i > r_j) \land (s_i > s_j)\) or \((r_i < r_j) \land (s_i < s_j)\); otherwise, it is discordant. According to the above definition, the explicit form of the Kendall tau rank correlation coefficient is

\[
\rho_K = \frac{2}{n(n-1)} \sum_{i<j} \text{sign}(r_i - r_j)\text{sign}(s_i - s_j),
\] (C11)

where \( \text{sign}(\cdot) \) represents the sign function.

Table I summarizes the correlation coefficients with \( n = 500 \). Specifically, when the number of supernets is 1, we have \( \rho_K = 0.113 \), which implies that the correlation between \( r \) and \( s \) is very low. By contrast, with increasing the number of supernets to 5 and 10, the correlation coefficients \( \rho_S \) and \( \rho_K \) are dramatically enhanced, which are 0.723 and 0.536, respectively. Moreover, when the number of supernets is \( W = 10 \) and the number of iterations is increased to \( T = 1000 \), the correlation coefficients \( \rho_S \) and \( \rho_K \) can be further improved, which are 0.774 and 0.591, respectively.

These results indicate that the competition phenomenon in QAS can be alleviated by introducing more supernets and increasing the number of training iterations. In doing so, the performance of subnets evaluated by QAS can well accord with their real performance with independently training.

| \( W = 1 \) & \( T = 500 \) | \( W = 5 \) & \( T = 500 \) | \( W = 10 \) & \( T = 500 \) | \( W = 10 \) & \( T = 1000 \) |
|---|---|---|---|
| \( \rho_S \) | 0.488 | 0.723 | 0.716 | 0.774 |
| \( \rho_K \) | 0.113 | 0.536 | 0.531 | 0.591 |

**TABLE I: The correlation coefficients.** The label ‘\( W = a \) \& \( T = b \)’ represents that the number of supernets and training iterations is \( a \) and \( b \), respectively.

### D. Experimental Details of the ground state energy estimation

In this section, we first briefly recap the ground state energy estimation task in Appendix D. In Appendix D, we compare the performance of QAS and conventional VQE towards the ground state energy estimation task when they are implemented on real quantum hardware.

#### 1. The ground state energy estimation

A central application of VQLS based algorithms is solving the electronic structure problem, i.e., finding the ground state energies of chemical systems described by Hamiltonians. Note that chemical Hamiltonians in the second quantized basis set approach can always be mapped to a linear combination of products of local Pauli operators [17]. In particular, the explicit form of the molecular hydrogen Hamiltonian \( H_h \) in Eqn. (7) is

\[
H_h = -0.042 + 0.178(Z_0 + Z_1) + 0.243(Z_2 + Z_3) + 0.171Z_0Z_1 + 0.123(Z_0Z_2 + Z_1Z_3) + 0.168(Z_0Z_3 + Z_1Z_2) + 0.176Z_2Z_3 + 0.045(Y_0X_1X_2Y_3 - Y_0Y_1X_2X_3 - X_0X_1Y_2Y_3 + X_0Y_1Y_2X_3).
\] (D1)

The goal of the variational Eigen-solver (VQE) is generating a parameterized wave-function \( \left| \Psi(\theta) \right\rangle \) to achieve

\[
\min_\theta |\left\langle \Psi(\theta) \right| H_h |\Psi(\theta) \right\rangle - E_m|.
\] (D2)
The linear property of $H_h$ in Eqn. [D1] implies that the value $\langle \Psi(\theta)|H_h|\Psi(\theta)\rangle$ can be obtained by iteratively measuring $|\Psi(\theta)\rangle$ using Pauli operators in $H_h$, e.g., such as $\langle \Psi(\theta)|\mathbb{I}_3 \otimes Z_0|\Psi(\theta)\rangle$ and $\langle \Psi(\theta)|X_0Y_1Z_2|\Psi(\theta)\rangle$. The lowest energy of $H_h$ equals to $E_m = -1.136$ Ha, where ‘Ha’ is the abbreviation of Hartree, i.e., a unit of energy used in molecular orbital calculations with $1$ Ha $= 627.5$ kcal/mol. The exact value of $E_m$ is acquired from a full configuration-interaction calculation.

We note that the quantum natural gradient optimizer [45], which can accelerate the convergence rate, is employed to optimize the trainable parameters for both VQE and QAS, where the learning rate is set as 0.2.

2. The performance of QAS on real quantum devices

Here we carry out QAS and the conventional VQE on IBM’s 5-qubit quantum machine, i.e., ‘Ibmq_ourense’, to accomplish the ground state energy estimation of $H_h$. The qubit connectivity of ‘Ibmq_ourense’ is illustrated in Figure 7 and the system parameters of these five qubits are summarized in Figure 8.

![FIG. 7: The qubits connectivity of ‘Ibmq_ourense’.

The implementation detail is as follows. The hyper-parameters of QAS are $L = 3$, $W = 10$, $K = 500$, and $T = 500$. To examine the compatibility of QAS, we restrict its searching spaces to be consistent with the qubit connectivity of ‘Ibmq_ourense’, i.e., the single qubit gates are sampled from $R_Y$ and $R_Z$, and CNOT gates can conditionally apply to the qubits pair $(0, 1)$, $(1, 0)$, $(1, 2)$, $(2, 1)$, $(1, 3)$, and $(3, 1)$, based on Figure 8. We call this setting as QAS with the real connectivity (QAS-RC). Under such a setting, the number of all possible circuit architectures for QAS-RC is 1024$. The hyper-parameters setting for VQE are $L = 3$ and $T = 500$. The heuristic circuit architecture used in VQE is identical to the case introduced in the main text (Figure 3 (a)). In the training process, we optimize VQE and QAS on classical computers under a noisy environment provided by the Qiskit package, which can approximately simulate the quantum gates error and readout error in ‘Ibmq_ourense’. The reason that we move the training stage on the classical simulators is because training VQE and QAS on ‘Ibmq_ourense’ will take an unaffordable runtime, due to the fair share run mode.

The training performance of VQE and QAS-RC is demonstrated in Figure 9. In particular, as shown in the left panel, the estimated ground energy by VQE is around $-1.02$ Ha after 30 iterations, highlighted by the dark blue line. The performance of QAS-RC is shown in the right panel. Concretely, when we retrain the output subnet with 15 iterations, its estimated energy slightly oscillates around $-1.04$ Ha, highlighted by the green solid line. When we implement the optimized VQE and the optimized output subnet of QAS-RC on the real quantum device, i.e., ‘Ibmq_ourense’, their performances are varied. Specifically, as demonstrated in the left panel of Figure 9, the estimated ground energy by VQE is $-0.61$ Ha (highlighted by the blue marker), while the estimated ground energy by QAS-RC is $-0.963$ Ha (highlighted by the green marker). Compared with VQE, the estimated result of QAS-RC is much closer to the exact result. We utilize the following formula to quantify the relative deviation between the simulation and experiment results. Denoted the estimated energy obtained by the numerical simulation as $E_s$ and the test energy achieved by ‘Ibmq_ourense’ as $E_t$, the relative deviation follows

$$\text{err} = \frac{|E_s - E_t|}{E_m},$$

where $E_m = -1.136$ is the exact result. Following this formula, the relative deviation for VQE and QAS-RC is 36.1% and 6.8%, respectively. Compared with the heuristic circuit architecture used in VQE, QAS that concerns
FIG. 9: Experiment results of the ground state energy estimation of VQE and QAS. In the left panel, the labels ‘Exact’ and ‘Energy_VQE’ correspond to the exact ground state energy and the estimated energy of VQE obtained in the training process (achieved by numerical simulators), respectively. In the left panel, the label ‘Energy_QAS_RC’ refer to the estimated energy of QAS-RC in the retraining phase. The estimated energy of VQE and QAS-RC when they are implemented on the real quantum processor ‘Ibmq_ourense’ are indicated by two cross markers, where the corresponding labels are ‘Energy_VQE_IBM’ and ‘Energy_QAS_RC_IBM’, respectively.

the real qubits connectivities can dramatically reduce the relative deviation. The above results not only indicate the compatibility of QAS, but also demonstrate that QAS can well adapt to the weighted gates noise and achieve a high performance towards quantum chemistry tasks.

FIG. 10: The simulation results of QAS-RC in the ranking stage. The left column demonstrates the subnets ranking distributions with $K = 500$ of QAS-RC. The x-axis refers to the estimated energy of the given subnet is in the range of $(a \text{ Ha}, b \text{ Ha}]$ with $a, b \in \mathbb{R}$. For most subnets, their estimated energies are above $-0.2 \text{ Ha}$. The middle column exhibits the output subnet of QAS-RC. The right column shows the implementation of the output subnet of QAS-RC on ‘Ibmq_ourense’.

Finally, we compare the output subnet of QAS-RC with the heuristic circuit architecture used in VQE. The simulation results of QAS-RC in the ranking stage are summarized in Figure 10. In particular, the left column exhibits the ranking distributions of QAS-RC, where the estimated ground energy of most subnets concentrates on $[-0.6 \text{ Ha}, -0.4 \text{ Ha}]$. The middle column of Figure 10 shows the output subnet of QAS-RC, where the corresponding circuit implementation on ‘IBM_ourense’ is exhibited in the right column. Compared with the heuristic circuit architecture used in VQE (Figure 3 (a)), the output subnet of QAS-RC contains fewer CNOT gates. This implies that QAS-RC has the ability to appropriately reduce the number of two qubits gate to avoid introducing too much error, while the expressive power of the trainable circuit $U(\theta)$ can be well preserved. In other words, QAS can adapt to the weighted gate noise to seek the best circuit architecture.

E. Improving the ranking stage of QAS

Recall the ranking stage of QAS, i.e., Step 3 of Figure 1, is uniformly sampling $K$ subnets from the supernet $A$. The aim of this step is sampling the one, among the sampled subnets, with the best performance. However, the uniformly sampling method implies that the sampled subnets maybe come from $S_{bad}$ with a high probability when $|S_{bad}| > |S_{good}|$. It is highly desired to devise more effective sampling methods.

Here we utilize an evolutionary algorithm, i.e, nondominated sorting genetic algorithm II (NSGA-II) [53], to facilitate the subnets ranking problem. The intuition behind employing NSGA-II is actively searching potential subnets with good performance instead of uniformly sampling subnets from all possible circuit architectures.

We apply QAS with the evolutionary algorithm to tackle the ground state energy estimation problem described in
the main text. Note that all hyper-parameters settings are identical to the uniformly sampling case, except for the settings related to the evolutionary algorithm. Particularly, we set the population size as $N_{\text{pop}} = 50$ and the number of generations as $G_T = 20$. The simulation results under the noiseless setting are shown in Figure 11. In particular, QAS assisted by NSGA-II searches in total 943 subnets, and the estimated energy of 143 subnets (15.2%) lies in the range from $-1 \text{ Ha}$ to $-1.2 \text{ Ha}$. By contrast, QAS with uniformly sampling strategy only finds 3 subnets among in total 500 subnets (0.6%) in the same range. This result empirically confirms that evolutionary algorithms can advance the performance of QAS.

![Simulation result of QAS assisted by NSGA-II](image)

**FIG. 11: Simulation result of QAS assisted by NSGA-II.** The label ‘QAS-no-evo, # E=5’ and ‘QAS-evo, # E=5’ refer to the QAS introduced in the main text and QAS assisted by evolutionary algorithm with the number of supernets being $W = 5$, respectively. The x-axis refers that the estimated energy of the given subnet is in the range of $(a \text{ Ha}, b \text{ Ha}]$ with $a, b \in \mathbb{R}$.

We remark that other advanced machine learning techniques such as reinforcement learning [54] can also be exploited to benefit the performance of QAS.