QuantumDARTS: Differentiable Quantum Architecture Search for Variational Quantum Algorithms

Wenjie Wu[1] Ge Yan[1] Xudong Lu[1] Kaisen Pan[1] Junchi Yan[1]

Abstract

With the arrival of the Noisy Intermediate-Scale Quantum (NISQ) era and the fast development of machine learning, variational quantum algorithms (VQA) including Variational Quantum Eigensolver (VQE) and quantum neural network (QNN) have received increasing attention with wide potential applications in foreseeable near future. We study the problem of quantum architecture search (QAS) for VQA to automatically design parameterized quantum circuits (PQC). We devise a differentiable searching algorithm based on Gumbel-Softmax in contrast to peer methods that often require numerous circuit sampling and evaluation. Two versions of our algorithm are provided, namely macro search and micro search, where macro search directly searches for the whole circuit like other literature while the innovative micro search is able to infer the sub-circuit structure from a small-scale and then transfer that to a large-scale problem. We conduct intensive experiments on unweighted Max-Cut, ground state energy estimation, and image classification. The superior performance shows the efficiency and capability of macro search, which requires little prior knowledge. Moreover, the experiments on micro search show the potential of our algorithm for large-scale QAS problems.

1. Introduction

Over the last decade, quantum mechanics as well as quantum computing has received increasing attention and numerous studies have been conducted to seek potential quantum supremacy. Quantum algorithms have demonstrated their supremacy over classical ones in certain fields, e.g. Shor (Shor, 1994) for integer factorization, QAOA (Farhi et al., 2014) for combinatorial optimization, UCCSD (Romero et al., 2017) for ground state energy estimation, etc. Recently, automatically designing a parameterized quantum circuits (PQC) for VQAs has drawn increasing attention (Grimsley et al., 2019; Ostaszewski et al., 2021; Zhang et al., 2022; Du et al., 2022; Wang et al., 2022a) which greatly reduces the labour of human experts and finds better PQCs for certain VQA problems.

The problem of QAS for VQA in this paper can be formulated as follows. Given a candidate quantum gate set $\mathcal{G}$, we find the best composition in the form of PQC and its corresponding unitary $\hat{\mathbf{U}}(\mathcal{A}, \theta)$, which minimizes the loss of the original VQA problem. Here $\mathcal{A}$ is the optimal circuit, θ is the best rotation parameters. $\mathbf{U}$ denotes the unitary transformation for the circuit and can be calculated by Eq. 1.

One could further consider two VQA-related settings of QAS by the criterion that if there exist input data samples or not: **i) search for VQE (without input data)**, namely Variational Quantum Eigensolver: it evolves the expectation value of the quantum system w.r.t. an observable, often a Hamiltonian, with a classical optimizer. **ii) search for QNN (with input data and ground truth)**, namely quantum neural network: given a dataset with samples $\mathbf{x}$ and the corresponding labels y in a way of supervised learning. This is akin to Neural Architecture Search (NAS) (Elsken et al., 2019) or more broadly speaking, AutoML on classic computers. In this case, the goal in search directly refers to the loss e.g. cross-entropy for a supervised quantum classifier (or an unsupervised one when labels are unavailable).

Neural Architecture Search (NAS), which automatically generates good neural network architectures (with parameters for training) without human labor, has achieved wide success in real-world problems, especially in vision e.g. image classification (Liu et al., 2018) and object detection (Wang et al., 2022c). QAS methods have long been inspired by NAS techniques and imitated their classical counterparts, e.g. the adoption of evolutionary algorithms (EA) (Las Heras et al., 2016) and reinforcement learning (RL) (Ostaszewski et al., 2021). Notably, the differentiable architecture search paradigm (DARTS) has not been adopted to QAS although it is dominantly popular in NAS literature (Liu et al., 2018). Instead of searching over a discrete

[1] MoE Key Lab of AI, Shanghai Jiao Tong University, Shanghai, China. Correspondence to: Junchi Yan <yanjunchi@sjtu.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

set of candidate architectures, DARTS relaxes the search domain to a continuous one to optimize the architecture (as well as trainable parameters) by gradient descent. However, QAS requires an extra physical constraint that the normalized weighted summation of the candidate gate unitary matrices must be unitary, while the Softmax scheme for updating architecture weights in DARTS literature cannot guarantee this property. One pertinent work that achieves differentiable search with both architecture weights $\mathcal{A}$ and rotation parameters θ for QAS refers to (Zhang et al., 2022), whereby Monte Carlo sampling is used to estimate the gradient by sampling multiple circuits at each epoch to approximate the continuous distribution of the architecture weights, which we believe is less efficient.

To achieve higher efficiency, in this paper we resort to the Gumbel-Softmax technique (Gumbel, 1954; Bengio et al., 2013; Jang et al., 2017) for circuit sampling to fill the above gap in QAS literature. This trick has been well adopted in machine learning literature including its use in inference-stage neural network speedup (Luo, 2017; Herrmann et al., 2020) and architecture design (Dong & Yang, 2019), etc. Specifically, we first sample a circuit based on the architecture weights and the Gumbel distribution. Then we update the rotation parameters in the circuit. We iteratively update the architecture weights and the rotation parameters until convergence. With the help of Gumbel-Softmax, we can use argmax to sample one candidate gate at a time during the forward propagation and update the architecture weights by Softmax during the backward propagation. This can guarantee the unitary property of the generated quantum circuit and greatly reduce the search time.

Perhaps more interestingly, for scalability, we propose a new paradigm that performs sub-circuit search and then stacks them into a complete quantum circuit. This is in contrast to many existing QAS literature (Du et al., 2022; Ostaszewski et al., 2021) on directly searching on the whole quantum circuit which may suffer from scalability issue as the search space increases exponentially when the circuit becomes larger. We term these two search schemes by micro search and macro search as shown in Fig. 1.

In line with the QAS literature (Zhang et al., 2022), our experiments are conducted under a full amplitude simulation written in Python, on three representative VQA tasks for macro search. The first two i.e. Unweighted Max-Cut and ground state energy estimation of molecules are well-studied VQE problems. Our method nails the unweighted Max-Cut problem on graphs with 10 nodes and achieves admissible results (within the chemical accuracy) for ground state energy estimation, with much fewer gates and shallower circuits compared to UCCSD (Romero et al., 2017). We also test our algorithm on image classification on MNIST (LeCun et al., 1998) in both noiseless and noisy environments. Our method outperforms classical CNN and QCNN (Hur et al., 2022). **The highlights of this paper are:**

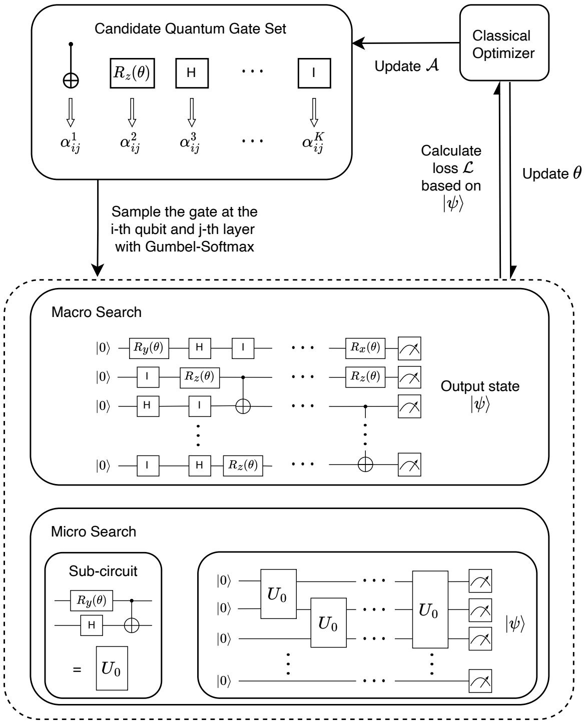


Figure 1. The pipeline of the proposed QuantumDARTS. First, we sample quantum gates from the candidate set with Gumbel-Softmax to achieve architecture search differentiability as well as the unitary property. The sampling probability of each gate is associated with α_{ij}. We denote the sampling coefficients of all the candidate gates by $\mathcal{A} = \{\alpha_{ij}\}$, which is also the architecture weights. Then, for macro search, we use the sampling results to construct the circuit directly. For micro search, we just sample a sub-circuit and then constitute the circuit with sub-circuits according to some predefined rules. Finally, we calculate the loss based on the output state $|\psi\rangle$, and update the architecture $\mathcal{A}$'s weights and rotation parameters θ with gradient descent.

1) We present an efficient QAS algorithm QuantumDARTS for VQA problems. By adopting Gumbel-Softmax for circuit sampling, we can efficiently update the architecture weight by gradient descent.

2) We propose two separate versions of our algorithm, namely macro search and micro search. Macro search aims to directly search for the whole circuit and micro search searches for quantum sub-circuits efficiently which are further stacked to form the whole circuit.

3) Experimental results on two VQE tasks and one QNN task verify the superior performance and cost-efficiency of our macro search compared with state-of-the-art QAS methods (Du et al., 2022; Zhang et al., 2022) and manual

designs. Moreover, the experiments on micro search show the potential for large-scale QAS problems.

2. Related Work

2.1. Quantum Architecture Search Algorithms

Evolutionary Algorithms. EA, especially genetic algorithms (GA), has been used to automatically design quantum circuits since (Williams & Gray, 1999). Numerous literature (Massey et al., 2004; Bang & Yoo, 2014; Las Heras et al., 2016; Lamata et al., 2018; Potoček et al., 2018) have demonstrated that GA is capable of evolving simple quantum circuits, such as quantum error correction code or quantum adder. GA requires a genetic representation of the solution domain and a fitness function to evaluate the solutions. The performance is severely affected by the gene length and the size of the candidate gate set. Moreover, it can not handle parameterized rotation gates, which hinders GA from being widely used in evolving VQA ansätze.

Reinforcement Learning Algorithms. The action in QAS is to select a gate from the candidate gate set and the environment is the whole circuit. The reward is used to train the agent to pick the right gates. One of the signature work in this track is (Ostaszewski et al., 2021). The authors focus on the VQE problems and test their algorithm on the ground state energy estimation problem on the 4-qubit and 6-qubit Hamiltonians of LiH. The circuit depth and number of gates used by the algorithm are much less than UCCSD (Romero et al., 2017) and the results are within the chemical accuracy. The works (Ye & Chen, 2021; Kuo et al., 2021) also use RL to search for quantum circuits. However, these two works only tested their algorithms on 2-qubit Bell state and 3-qubit GHZ state with circuit noise.

Sampling Based Learning Algorithm. (Grimsley et al., 2019; Tang et al., 2021) introduce adaptive VQE ansatz search for ground state energy estimation. (Du et al., 2022) searches for a quantum ansatz, which applies single-qubit gate layers and two-qubit gate layers alternatively. It samples a circuit from the candidate set and then updates the rotation parameters. (Zhang et al., 2022) uses Monte Carlo sampling to sample circuits from the candidate set and also searches for circuits based on task-wise predefined circuit structures. The algorithm is tested on QFT and Max-Cut problems. However, sampling from the candidate set is indifferentiable (except for Monte Carlo sampling) and inefficient. It also requires predefined circuit structures that limit the applicability to an unseen quantum ansatz. In this paper, we borrow the idea of Gumbel-Softmax to achieve quantum meaningful differentiable search of architectures without specific search space prior. We summarize the properties of some QAS methods in Appendix G.1.

2.2. (Differentiable) Neural Architecture Search

In this section, we give a brief review of the classical NAS approaches in vision, especially those using differentiable sampling (Liu et al., 2018; Dong & Yang, 2019). Differentiable neural architecture search (DARTS) is proposed by (Liu et al., 2018), which has been further refined with many critical improvements and generalizations. The key idea of DARTS is to relax the discrete search space of the neural architectures to a continuous domain and then end-to-end optimize the target differentiably. Other dominant approaches, such as RL (Zoph et al., 2018), EA (Real et al., 2019), and Bayesian Optimization (Kandasamy et al., 2018) are often less efficient since they treat NAS as a black-box optimization over a discrete domain, which leads to a huge number of architecture evolution. Beyond DARTS, there are recent works in the between. In (Wang et al., 2022b), the zero-order architecture search (ZARTS) is developed which achieves efficient zero-order optimization for NAS. For DARTS, when updating the architecture weights, it specifically uses Softmax for both forward and backward propagation. There are improvements as variants of DARTS. The GDAS (Dong & Yang, 2019) introduces Gumbel-Softmax technique to accelerate the searching process, which results in argmax during the forward and Softmax during the backward propagation. To stabilize the searching process, a skip connection suppression scheme is devised in (Chu et al., 2021). To reduce the memory cost for maintaining the search space in DARTS, the operation merge scheme is developed in (Wang et al., 2020).

3. Method

3.1. Approach Overview and Key Idea

As shown in Fig. 1, we recall the problem formulation of QAS in the beginning of Sec. 1: it seeks an optimal quantum circuit with its unitary $\hat{\mathbf{U}}(\mathcal{A}, \theta)$ from a candidate gate set $\mathcal{G}$. For VQE when only the observable Hamiltonian is given, the goal is to directly minimize the expectation. For the case of QNN when only training data samples are given, it is more akin to the NAS case by minimizing the loss $\mathcal{L}$ based on the output quantum state and the ground truth of the training samples. Note that in our implementation, we also consider the multi-qubit gates in $\mathcal{G}$, e.g. Toffoli, CNOT or CZ, which require a target qubit and multiple control qubits. Taking two-qubit gate CZ as an example, $\mathcal{G} = \{\text{CZ}\}$ is an abbreviation of $\mathcal{G} = \{\text{CZ}_1, \cdots, \text{CZ}_{n-1}\}$, which means each two-qubit gate will add $n-1$ candidates to $\mathcal{G}$. Without loss of generality, we define the qubit which picks CZ from $\mathcal{G}$ as the target qubit and the control qubit is determined by the subscript number, which maps to all the qubits except for the target iteratively.

For an n-qubit m-layer quantum circuit, we denote the se-

Algorithm 1 Macro quantum architecture search

```
Require: Randomly initialized $\mathcal{P}$, $\mathcal{Q}$ and $\theta$, training epoch $num\_epoch$, iteration number $num\_iter$
  Initialize $\hat{\mathbf{U}} \leftarrow I_{2^n}$;
  for $epoch \leftarrow 1$ to $num\_epoch$ do
    for $layer \leftarrow 1$ to $m$ do
      for $qubit \leftarrow 1$ to $n$ do
        Obtain $\hat{\mathbf{U}}_{qubit,layer}$ based on Eq. 3;
        $\hat{\mathbf{U}} := \hat{\mathbf{U}}_{qubit,layer} \times \hat{\mathbf{U}}$;
      end for
    end for
    for $iter \leftarrow 1$ to $num\_iter$ do
      Calculate loss $\mathcal{L}_\theta$;
      Update $\theta$ by gradient descent: $\theta := \theta - \nabla_\theta \mathcal{L}_\theta$;
      Update $\hat{\mathbf{U}}$ with the updated $\theta$;
    end for
    Calculate loss $\mathcal{L}_\mathcal{A}$;
    Update $\mathcal{P}$ by gradient descent: $\mathcal{P} := \mathcal{P} - \nabla_\mathcal{P} \mathcal{L}_\mathcal{A}$;
    Update $\mathcal{Q}$ by gradient descent: $\mathcal{Q} := \mathcal{Q} - \nabla_\mathcal{Q} \mathcal{L}_\mathcal{A}$;
  end for
  Fix the final circuit from $\mathcal{A}$ and optimize parameters $\theta$.
```

lection of candidate gates as a matrix $\mathbf{M} \in \mathcal{G}^{n \times m}$. Define $\mathbf{U}_{ij} = \sigma(\mathbf{M}_{ij})$ which maps a quantum gate on the i-th qubit and the j-th layer to a $2^n \times 2^n$ unitary with all the irrelevant qubits as identity. The unitary matrix of this circuit is

$$\hat{\mathbf{U}} = \prod_{j=1}^{m} \prod_{i=1}^{n} \hat{\mathbf{U}}_{ij} = \prod_{j=1}^{m} \prod_{i=1}^{n} \sigma(\mathbf{M}_{ij}) \tag{1}$$

For the position on the i-th qubit and the j-th layer, $\mathbf{M}_{ij}$ is sampled from a discrete probability distribution $\mathcal{T}_{ij}$, which is deduced from the probability mass function:

$$\mathcal{T}_{ij}^{(k)} = \Pr\left(\mathbf{M}_{ij} = \mathcal{G}^{(k)}\right) = \frac{\exp(\alpha_{ij}^{(k)})}{\sum_{k'=1}^{K} \exp(\alpha_{ij}^{(k')})}, \tag{2}$$

where $\alpha_{ij}^{(k)}$ represents the k-th element of the parameter $\alpha_{ij} \in \mathbb{R}^K (K = |\mathcal{G}|)$, and $\mathcal{G}^{(k)}$ denotes the k-th gate in $\mathcal{G}$. Notice that the back-propagation for gradients of α_{ij} cannot apply to sampling from the discrete distribution $\mathcal{T}_{ij}$. Therefore, we introduce the Gumbel-Softmax to allow back-propagation (Dong & Yang, 2019):

$$\begin{aligned} \hat{\mathbf{U}}_{ij} = \sigma(\mathbf{M}_{ij}) = \sum_{k=1}^{K} h_{ij}^{(k)} \sigma(\mathcal{G}^{(k)}) \\ \text{s.t. } h_{ij} = \text{one-hot}(\arg\max_k(\alpha_{ij}^{(k)} + G_k)) \end{aligned} \quad , \tag{3}$$

where G_k is a set of random variables sampled from the Gumbel distribution $G = -\log(-\log(X))$ with $X \sim U(0,1)$ (Chang et al., 2019). However, this sampled h_{ij} cannot be directly used in the neural network since $\arg\max$ is not differentiable. Gumbel-Softmax treats the problem by sampling h_{ij} as Eq. 3 in the forward process and substitutes $\arg\max$ with Softmax to enable gradient back-propagation in the backward process, during which the Gumbel-Softmax estimation is (Dong & Yang, 2019):

$$\tilde{h}_{ij}^{(k)} = \frac{\exp\left(\left(\log(\mathcal{T}_{ij}^{(k)}) + G_k\right)/\tau\right)}{\sum_{k'=1}^{K} \exp\left(\left(\log(\mathcal{T}_{ij}^{(k')}) + G_{k'}\right)/\tau\right)}, \tag{4}$$

where τ is the temperature. When τ goes to zero, $\tilde{h}_{ij}^{(k)}$ will tend to $h_{ij}^{(k)}$. When τ increases, the approximated distribution will gradually converge to a uniform distribution. Introducing Gumbel-Softmax into the searching network can fill the gaps between discrete sampling and gradient back-propagation, which has been an infamous obstacle blocking quantum arbitrary unitary approximation. Only one gate is chosen at each step, ensuring the physical meaning of the sampled circuit.

To further enlarge the parameter space and stabilize the searching process, we decompose the K dimensional vector α_{ij} into two matrix denoted as $\mathbf{P}_{ij} \times \mathbf{Q}_{ij}$, where $\mathbf{P}_{ij} \in \mathbb{R}^{1 \times K'}$ and $\mathbf{Q}_{ij} \in \mathbb{R}^{K' \times K}$ (K' is a predefined hyperparameter). The number of parameters for position ij increases from K to $K' \times (K+1)$, which leads to better performance of our model (detailed ablation study is shown in Appendix H). We denote $\mathcal{P} = \{\mathbf{P}_{ij}\}$ and $\mathcal{Q} = \{\mathbf{Q}_{ij}\}$.

We further propose two different versions of our algorithm.

3.2. QuantumDARTS: Macro Search

Macro search as most peer QAS methods do, aims to directly discover the entire quantum circuit from the candidate gate set. With the expansion of the circuit size, macro search will be much more difficult since the search space will increase exponentially. Unlike previous QAS works (Zhang et al., 2022; Du et al., 2022) which usually have predefined circuit structures, we have very little specific prior knowledge for the macro search except for the basic and general search space[1]: for an n-qubit m-layer circuit with K candidate gates, and the search space for our macro search is $K^{n \times m}$.

The objective is to find a certain quantum circuit that minimizes the loss $\mathcal{L}$ which varies according to the setting of VQE and QNN as discussed in Sec. 3.1. By utilizing Eq. 4, we can make sure the sampling procedure is differentiable and we can learn a distribution of the candidate gate set $\mathcal{G}$. The circuit architecture weights $\mathcal{P}$, $\mathcal{Q}$ and rotation parameters θ are updated iteratively as given in Alg. 1. For each

[1] In contrast to the NAS literature where search space (SS) have been well developed, the SS is more subtle and not well defined in QAS literature, and we here adopt a general SS. This also makes the search technique comparison independent from SS.

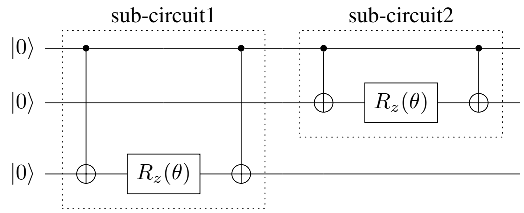


Figure 2. Sub-circuits in QAOA (Farhi et al., 2014). Sub-circuit1(2) is matched with edge $e_{13}(e_{12})$. Hence, the gates in it only function on qubit 1 and 3(1 and 2). Other sub-circuits share the same structure but act on different pairs of qubits.

epoch, we sample a circuit based on h_{ij} and calculate the overall unitary $\hat{\mathbf{U}}$ of this circuit. We obtain the loss $\mathcal{L}_\theta$ and $\mathcal{L}_\mathcal{A}$ from the loss customized to each task. The rotation parameters θ are updated for $iter\_num$ iterations to guarantee that we minimize the loss for this certain circuit architecture and then we update the architecture parameters $\mathcal{P}$ and $\mathcal{Q}$. The complexity of our algorithm is analyzed in Appendix B.

3.3. QuantumDARTS: Micro Search

Micro search aims to discover quantum sub-circuits and designs the overall quantum circuit by stacking many copies of the discovered sub-circuits. Introducing micro search is very important for searching for large-scale quantum circuits. The essence of micro search is to infer the structure of sub-circuits in a small-scale problem, and then share the architecture weights for the sub-circuits and apply it multiple times to a large-scale problem. The QAOA circuit is a typical example for micro search. As shown in Fig. 2, each edge e_{ij} which connects node v_i and v_j maps to a sub-circuit on the i-th qubit and the j-th qubit. The sub-circuit structure is an RZZ gate (CNOT-R$_z$(θ)-CNOT), which is only a 2-qubit 3-layer circuit. Searching for this sub-circuit and then applying the sub-circuit multiple times based on the graph structure is much easier than directly searching for the whole circuit.

Given an $n \times m$ circuit with the sub-circuit of size $n' \times m'$ ($n' < n$, $m' < m$). The search space for micro search is reduced to $k^{n' \times m'} (k < K)$. We have a predefined rule that indicates where to apply the sub-circuits. During the training process, we have only one group of architecture weights and we can have multiple groups of rotation parameters for different sub-circuits.

4. Macro Search Experiments

Our proposed model is tested in three representative VQA tasks in recent literature, i.e. ground state energy estimation, unweighted Max-Cut, and image classification. Besides manual designs, we also compare our model with DQAS (Zhang et al., 2022), QCAS (Du et al., 2022), and random sampling (RS) in the first and third task. RS is based on our method with the architecture randomly sampled initially and then fixed.

Experiments are performed on a commodity workstation with 4 CPUs with 224 cores Intel(R) Xeon(R) Platinum 8276 CPU @ 2.20GHz, and a GPU (NVIDIA A100 PCIe). The source code is written using PyTorch v1.12.1. The candidate set $\mathcal{G}$ is set as {R$_z$R$_y$R$_z$, I, CNOT} in all the experiments for its strong expressivity (see proof in Appendix C). We use the Adam optimizer and a cosine annealing schedule (Loshchilov & Hutter, 2016) to train our model.

4.1. Task I: Ground State Energy Estimation (VQE)

4.1.1. Background

In quantum chemistry, the ground state of a molecule is its stationary state with the lowest allowed energy, i.e. ground state energy E_0, which can be estimated given the types and relative coordinates of its atoms. This energy is closely related to the molecular Hamiltonian $\mathcal{H}_m$, which is an operator embodying the energy of the electrons and nuclei in a molecule. Given the state vector $|\psi\rangle$, we have:

$$E_0 \leq \frac{\langle\psi| \mathcal{H}_m |\psi\rangle}{\langle\psi|\psi\rangle}, \tag{5}$$

where the equality holds if and only if $|\psi\rangle$ is the ground state. In fact, the ground state energy is the smallest eigenvalue of the molecular Hamiltonian and the corresponding eigenvector is the ground state (Richard & David, 1993). With the Hartree-Fock method (Slater, 1951), we can approximate the molecular Hamiltonian according to the types and relative coordinates of atoms in a molecule.

To accelerate the estimating process and also improve accuracy, manually-designed VQE ansätze have been proposed in the literature, e.g. the hardware-efficient ansatz (Kandala et al., 2017) and the UCCSD ansatz (Hoffmann & Simons, 1988; Bartlett et al., 1989). The former fixes its architecture as sequential rotation gates and two-qubit entangling gates, while the latter designs a specialized ansatz for each molecule based on the unitary coupled cluster (UCC) theory. Specifically, UCCSD utilizes Trotter-Suzuki decomposition to generate the circuit, which is extremely large and could be practically intractable. Here we aim to obtain an estimated value of E_0 within the chemical accuracy yet at the cost of fewer gates. The chemical accuracy, approximately 0.0016 Hartree (Ha), is essential for realistic chemical predictions.

4.1.2. Experiment Setting

We use the output of the quantum circuit as the state vector $|\psi\rangle$. By convention, the input state of quantum circuits is set to $|\psi_0\rangle = |0\rangle^{\otimes n}$, where n is the number of qubits. We denote the matrix of the quantum circuit as $\mathbf{U}_m(\mathcal{A}, \theta)$.

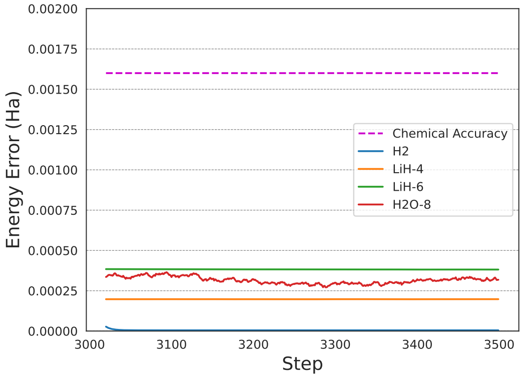


Figure 3. Learning curves of different molecules. Energy error means the gap between our estimated energy and the accurate ground state energy. We only depict a part of the training process, in which all learning curves reach relatively low values. The purple dashed line indicates the chemical accuracy.

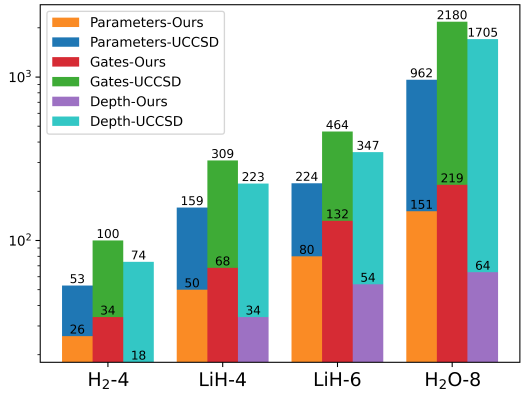


Figure 4. Comparison between the complexity of our model and the UCCSD ansatz. The depth and number of gates are measured through Qiskit. The number of parameters here equals the number of rotation gates in a circuit. The vertical axis is logarithmic.

Table 1. Comparison of energy errors in Hartree among different models. Energy errors within chemical accuracy are underlined.

MODEL	H_2	LiH-4	LiH-6	H_2O-8
UCCSD	$\underline{5.5 \times 10^{-11}}$	$\underline{4.0 \times 10^{-5}}$	$\underline{4.0 \times 10^{-5}}$	$\underline{4.0 \times 10^{-6}}$
OURS	$\underline{4.3 \times 10^{-6}}$	$\underline{1.7 \times 10^{-4}}$	$\underline{2.9 \times 10^{-4}}$	$\underline{3.1 \times 10^{-4}}$
QCAS	2.2×10^{-2}	8.6×10^{-2}	7.3×10^{-2}	7.0×10^{-1}
DQAS	$\underline{3.1 \times 10^{-4}}$	$\underline{5.3 \times 10^{-4}}$	$\underline{1.5 \times 10^{-3}}$	5.2×10^{-1}
RS	1.9×10^{-2}	1.3×10^{-2}	6.2×10^{-3}	4.0×10^{-1}

Then, we have $|\psi\rangle = \mathbf{U}_m(\mathcal{A}, \theta)\,|\psi_0\rangle$. Since every gate in the quantum circuit can be represented by a unitary matrix, $\mathbf{U}_m(\mathcal{A}, \theta)$ is also a unitary matrix, indicating that $\langle\psi|\psi\rangle = 1$. Hence, we rewrite Eq. 5 as:

$$E_0 \leq \langle\psi_0|\, \mathbf{U}_m^\dagger(\mathcal{A}, \theta)\mathcal{H}_m\mathbf{U}_m(\mathcal{A}, \theta)\,|\psi_0\rangle\,, \tag{6}$$

We formulate our loss function $\mathcal{L}(\mathcal{A}, \theta)$ as:

$$\mathcal{L}(\mathcal{A}, \theta) = \langle\psi_0|\, \mathbf{U}_m^\dagger(\mathcal{A}, \theta)\mathcal{H}_m\mathbf{U}_m(\mathcal{A}, \theta)\,|\psi_0\rangle\,. \tag{7}$$

Theoretically, we can approximate the ground state energy E_0 closely enough by minimizing $\mathcal{L}(\mathcal{A}, \theta)$.

Since the memory cost rises exponentially with the number of qubits, we only compute the ground state energy of three small molecules, i.e. hydrogen (H_2), lithium hydride (LiH) and water (H_2O). Given the types and relative coordinates of every atom in a molecule, we can derive its Hamiltonian easily with the Python package OpenFermion (McClean et al., 2020). The Hamiltonians of these three molecules require 4, 12 and 14 qubits, respectively. Following the protocol in peer literataure (Ostaszewski et al., 2021) to make the experiments within an affordable GPU memory cost, we reduce the scales of the Hamiltonians for H_2O from 14 to 8 qubits, and LiH from 12 to 6 and 4 qubits, at a trivial cost of accuracy. We denote these molecules as H_2, LiH-4, LiH-6 and H_2O-8. The relative 3D coordinates of these molecules are provided in Appendix D.1. To balance the accuracy and circuit complexity, we study the relationship between accuracy and the number of layers in Appendix D.2.

4.1.3. Results and Discussion

As shown in Fig. 3, the estimated energy of all four molecules reaches a low value. The gap between the estimated energy and the true ground state energy is far enough within chemical accuracy. Tbl. 1 shows the performance of different methods on ground state energy estimation. Our method surpasses peer QAS methods (DQAS (Zhang et al., 2022) and QCAS (Du et al., 2022)) in both accuracy and efficiency. Performance of some other QAS methods is discussed in Appendix G.2. Though UCCSD outperforms our model, the energy errors of our model are already far lower than the chemical accuracy 1.6×10^{-3} Ha, which is sufficient for realistic chemical predictions. Note that the reported energy errors may be a little different from results in Fig. 3, because the results in Fig. 3 may not be the lowest value in the whole training process.

We also compare the complexity of our model with the UCCSD ansatz (Romero et al., 2017) in Fig. 4. The UCCSD ansatz is constructed via the Python package Qiskit (Aleksandrowicz et al., 2019). Although the UCCSD ansatz can approach the ground state energy more closely, it demands a substantial number of gates. Especially given a large number of qubits, e.g. H_2O-8, the number of gates and the circuit depth of UCCSD are almost an order of magnitude larger than those of our model. On a quantum device, a deeper circuit will be more prone and vulnerable to circuit noise (Cerezo et al., 2021) and might lead to the phenomenon called quantum decoherence (Zeh, 1970). Our

Table 2. Statistics of our generated graphs by the Erdős–Rényi model with 10 vertices for Max-Cut. P_e is the probability of edge creation. $|\mathcal{E}|$ represents the number of edges. $\overline{D}$, D_{min} and D_{max} denote the mean, minimum and maximum degree, respectively.

P_e	$\|\mathcal{E}\|$	$\overline{D}$	D_{min}	D_{max}
0.25	12.6± 3.0	2.5± 1.3	0.6± 0.5	4.6± 1.4
0.50	22.1± 3.8	4.4± 1.6	2.5± 1.0	6.6± 0.8
0.75	33.5± 3.1	6.7± 1.4	5.0± 0.9	8.4± 0.5

method is capable of solving ground state energy estimation with more physically realizable quantum circuits.

4.2. Task II: Unweighted Max-Cut (VQE)

4.2.1. Background

The unweighted Max-Cut problem is a classical combinatorial optimization problem. Given the topology of an unweighted graph, we partition all the vertices into two complementary sets so that the number of edges between these two sets is maximized. We can formulate this problem as:

$$\begin{aligned} \max \quad & \sum_{0<i<j\leq n} w_{i,j}\frac{1-x_i x_j}{2} \\ s.t. \quad & x_k \in \{-1,1\}, \quad 1\leq k\leq n \\ & w_{i,j} = \mathbb{1}_{e_{ij}\in\mathcal{E}}, \quad 1\leq i<j\leq n \end{aligned}, \tag{8}$$

where $\mathcal{E}$ is the edge set and n is the number of nodes in the graph. The statement $e_{ij} \in \mathcal{E}$ is equivalent to an edge linking node v_i with node v_j. This problem is proved to be NP-hard (Karp, 1972), which means that no polynomial-time algorithms for it are known. To leverage the quantum supremacy, quantum annealing is used to seek potential polynomial-time algorithms for the Max-Cut problem. A typical example is the quantum approximation optimization algorithm (QAOA) (Farhi et al., 2014).

Now we reformulate this problem in a quantum computing genre. Each graph in Max-Cut is matched with a Hamiltonian $\mathcal{H}_c$ and each qubit represents a node in the graph. The Hamiltonian can be calculated as:

$$\mathcal{H}_c = \sum_{e_{ij}\in\mathcal{E}} \frac{1}{2}\left(I_n - Z_n^{(i)} Z_n^{(j)}\right), \tag{9}$$

where I_n is an n-qubit identity matrix, and $Z_n^{(i)}$ indicates a Pauli-Z gate operating on the i-th qubit. The largest eigenvalue of $\mathcal{H}_c$ amounts to the maximal cut value.

4.2.2. Experiment Setting

Akin to ground state energy estimation, the loss function $\mathcal{L}(\mathcal{A},\theta)$ in Max-Cut is defined as:

$$\mathcal{L}(\mathcal{A},\theta) = -\langle\psi_h|\,\mathbf{U}_c^\dagger(\mathcal{A},\theta)\mathcal{H}_c\mathbf{U}_c(\mathcal{A},\theta)\,|\psi_h\rangle\,, \tag{10}$$

where $\mathbf{U}_c(\mathcal{A},\theta)$ is the matrix of our circuit, and the minus sign ensures that this is a minimization problem. Following QAOA, we set the input state $|\psi_h\rangle$ to $|+\rangle^{\otimes n}$, obtained by a layer of Hadamard gates H:

$$|\psi_h\rangle = |+\rangle^{\otimes n} = \mathrm{H}^{\otimes n}\,|\psi_0\rangle\,.$$

We will further explain how to derive the predicted partition from the output of our circuit in Appendix E.

We generate random graphs according to the Erdős–Rényi model (Erdős et al., 1960). We generate graphs with 10 vertices and vary the density of graphs by setting the probability of edge creation P_e as 0.25, 0.50 and 0.75, respectively. For each density, we sample 10 random graphs. Some statistics on the generated graphs are listed in Tbl. 2.

4.2.3. Results and Discussion

In the experiments, we fix the number of layers as 15 and repeat each experiments three times. Our model succeeds in finding the maximal cut values of all the generated graphs. We demonstrate an optimal solution found by our model for each P_e in Fig. 5. The performance of our model is consistent under different density of graphs. We refer to Appendix I for generated circuits of these three graphs. Besides, a brief comparison between other QAS method and ours is offered in Appendix G.3.

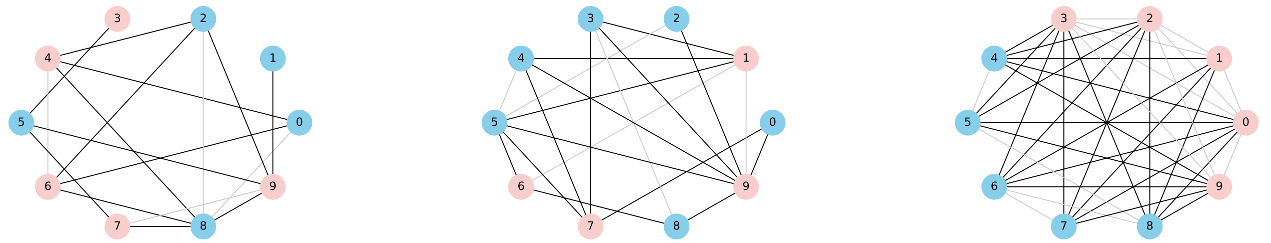
4.3. Task III: Image Classification (QNN)

4.3.1. Background

For image classification, given a set of images X and corresponding labels Y, the goal is to find a mapping f so that the prediction $\hat{Y} = f(X)$ is as close to Y as possible. For classical models, Convolutional neural networks (CNNs) like ResNet (He et al., 2016) are widely used. Attention mechanism (Vaswani et al., 2017) has also been adapted to image classification as proposed in Vision Transformer (ViT) (Dosovitskiy et al., 2020). Correspondingly, (Cong et al., 2019) proposes quantum convolutional neural networks (QCNNs), and (Hur et al., 2022) further evaluates it on the MNIST dataset. Following the setting in (Hur et al., 2022), we compare the performance of our model with both CNN and QCNN. MNIST contains hand-written digits from 0 to 9. Each sample is an image of 28×28 pixels.

4.3.2. Experiment Setting

Since we focus on binary classification, we screen out class 0 and class 1 from MNIST to form a new dataset. The training set and test set contain 12,665 and 2,115 samples, respectively. Following the pre-processing protocol in (Hur et al., 2022), we use Principal Component Analysis (PCA) or an autoencoder (AE) to reduce the original 28×28 image to an m-dimension vector $\mathbf{x}_i$ as the input.



(a) $P_e = 0.25$.

(b) $P_e = 0.50$.

(c) $P_e = 0.75$.

Figure 5. Optimal solutions of the Max-Cut problem found by our model via micro search under different edge creation probability P_e. All nodes in the graph are partitioned into two complementary sets (colored in pink and blue). Edges between these two complementary sets are colored in black, while others are colored in gray. The maximal cut values for 5a, 5b and 5c are 13, 15 and 24 respectively.

Table 3. Average numbers of used parameters under four settings for converting the raw image into an embedded vector.

SETTING	OURS	CNN	QCNN
PCA-ANGLE	37.8	44.0	51.0
AE-ANGLE	29.8	26.0	51.0
PCA-DENSE	30.2	56.0	51.0
AE-DENSE	35.4	34.0	51.0

We set $m = 8$ if using angle encoding, and $m = 16$ if using dense angle encoding. Details on the above autoencoder and data encoding schemes can be found in Appendix F. Supposing that the data input is $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(m)})$ (m is even) and the encoded state is $|\psi(\mathbf{x}_i)\rangle$, we then input $|\psi(x_i)\rangle$ into our model and obtain the output state $|\psi_t(\mathbf{x}_i)\rangle = \mathbf{U}_r(\mathcal{A}, \theta)\, |\psi(\mathbf{x}_i)\rangle$. Recalling that we target binary classification, we only need to measure the last qubit. The cross-entropy loss can be written by:

$$\mathcal{L}(\mathcal{A}, \theta) = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i \log \Pr\Big(\mathcal{M}\big(|\psi_t(\mathbf{x}_i)\rangle \big) = |1\rangle \Big) \\ + (1 - y_i) \log \Pr\Big(\mathcal{M}\big(|\psi_t(\mathbf{x}_i)\rangle \big) = |0\rangle \Big) \Big), \tag{11}$$

where y_i is the binary label of $\mathbf{x}_i$ and $\Pr\Big(\mathcal{M}\big(|\psi_t(\mathbf{x}_i)\rangle \big) = |k\rangle \Big)$ denotes the probability that the measured state of the last qubit is $|k\rangle$ ($k \in \{0, 1\}$). See derivation in Appendix F.

The compared CNN and QCNN are both from (Hur et al., 2022). We train CNN, QCNN and our model on the whole training set by 5 epochs and evaluate them on the test set. The parameters in each model are tuned to a similar amount for fair comparison. Specifically, the CNN is composed of two 2×2 1D convolutional layers (each followed by a ReLU activation and a 1D max pooling layer) and an FC layer.

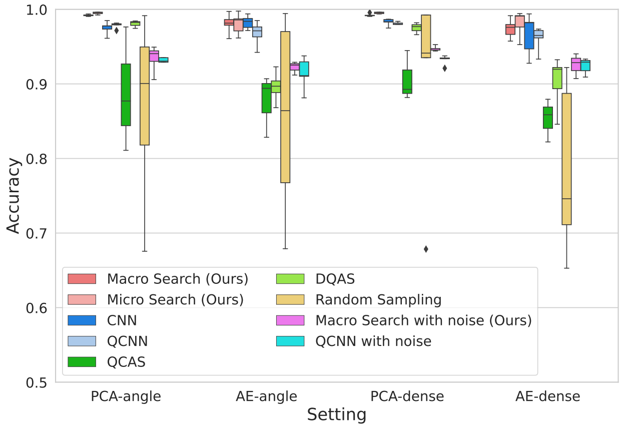


Figure 6. Accuracy on MNIST under different settings. Four settings use different combinations of dimension reduction and data encoding methods. Each experiment is repeated five times.

4.3.3. RESULTS AND DISCUSSION

As shown in Fig. 6, our model consistently outperforms QCNN and other QAS methods in all four settings by a large gap. Besides, our model significantly surpasses CNN under PCA-angle and PCA-dense settings, while it keeps competitive under the other two settings. It seems our model is more compatible with PCA and angle encoding. We refer to Appendix G.4 for comparison with more QAS methods.

We compare the average number of parameters in Tbl. 3. Our model uses fewer parameters than CNN and QCNN, but it still yields higher accuracy, verifying its cost-efficiency.

To test the performance under circuit noise, we add a 4.77% bit flip and a 4.77% phase flip as the readout noise (the error rate is obtained from Sycamore (Arute et al., 2019) and Zuchongzhi (Wu et al., 2021)). The depolarizing noise requires the density operator instead of quantum states, which greatly increases the memory cost to simulate the whole quantum process. Thus, we use the readout noise for now. Results in Fig. 6 showcases that our model still performs

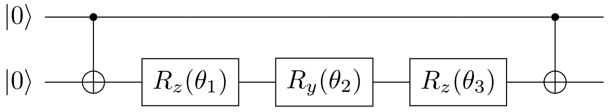


Figure 7. A sub-circuit found by our model resembles QAOA sub-circuits. Only 2 out of 10 qubits are depicted for simplicity. Identity gates are omitted since it makes no difference to the circuit.

better than QCNN in noisy environments.

5. Micro Search Experiments

Instead of directly searching for the macro circuit, we focus on its micro sub-circuits in this section. Inspired by QAOA, we try to utilize the intrinsic symmetry in the unweighted Max-Cut problem. Since every edge shares the same property (i.e. weight) and the only difference is the nodes they connect, we can safely assume that the sub-circuits encoding the edge information in the circuit share the same structure. In other words, every edge is mapped to a sub-circuit acting on two qubits matched with its two vertices. Then, each sub-circuit is stacked into the whole circuit one by one according to the edges in Max-Cut. A detailed example is given in Fig. 10. Besides, a layer of rotation gates $R_zR_yR_z$ is added at the end to improve the expressiveness of our model. Note that rotation parameters in sub-circuits vary. The results show that our model finds all the 5 distinct solutions for the graph in Fig. 5b and the predicted probability for each solution is divided equally, i.e. 0.2. Therefore, our model perceives the symmetry in Max-Cut and can find multiple optimal solutions simultaneously. Note the two swapped complementary sets are regarded as the same solution.

We use the same parameters in all the sub-circuits to enhance the symmetry in our model. To our surprise, similar sub-circuits are found in QAOA, which proves that our model can find promising sub-circuits with structures suited to the particular properties in the problem. Since sub-circuits share the parameters, the number of parameters is reduced from 59 to 6. We refer to Fig. 7 for more details.

We also test micro search on image classification to elucidate the mechanism of micro search. The micro search architecture for image classification is depicted in Fig. 8. We design a sub-circuit (denoted as U_{ij}) with two qubits (q_i and q_j) as input and the qubit with smaller index as the output for successors. This sub-circuit entangles two qubits and integrates their information into one qubit, which functions like the traditional convolutional layer and pooling layer. As for our experimental setting, the image data is encoded into 8 qubits. We apply 4 sub-circuits (U_{01}, U_{23}, U_{45}, U_{67}) in the first layer, 2 sub-circuits (U_{02}, U_{46}) in the second layer and 1 sub-circuit (U_{04}) at the last layer to form the entire circuit. All the 7 sub-circuits share the same architecture weights but have different rotation parameters. Finally, q_0 is measured to predict the label. Fig. 6 shows that micro search can achieve comparable accuracy to macro search.

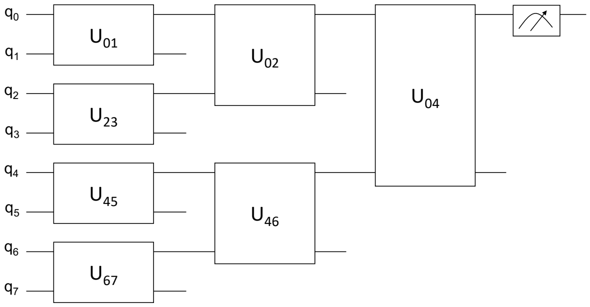


Figure 8. The architecture of micro search for image classification.

The advantage of micro search is inferring the structure of sub-circuits in small-scale problems with the properties of problems and then stacking them to construct the whole circuit in large-scale problems. The inferred sub-circuits can be similar to or even better than some existing manually designed ones like QAOA. Since architecture search in large-scale problems is more difficult and time-consuming, let alone manual design, our scheme can be a shortcut to avoid this issue, strengthening the scalability of our model.

6. Conclusion and Future Work

We have presented an end-to-end differentiable QAS algorithm via the Gumbel-Softmax technique, specifically with two versions, namely macro search and micro search. Experiments for macro search on three mainstream and well-studied VQA tasks demonstrate the efficiency and capability of our algorithm. Experiments for micro search on unweighted Max-Cut and image classification illustrate its potential scalability to infer the sub-circuit structure on small-scale and apply it to large-scale problems. Our algorithm is expected to be of practical value for automatically designing the quantum oracles as well as generating VQA ansätze, which are building blocks in the NISQ era.

Future Work. The QAS problem has various applications apart from the VQA problems. (Li et al., 2017; Deibuk & Biloshytskyi, 2015; Lamata et al., 2018) search for quantum oracles and (Rigby, 2021; Nautrup et al., 2019) utilize the QAS approaches to design quantum error correction codes. We would further evaluate our method on different applications and especially for large-scale quantum circuits.

Acknowledgments

The work was supported in part by National Key Research and Development Program of China (2020AAA0107600) and NSFC (62222607).

References

Aleksandrowicz, G., Alexander, T., Barkoutsos, P., Bello, L., Ben-Haim, Y., Bucher, D., Cabrera-Hernández, F. J., Carballo-Franquis, J., Chen, A., Chen, C.-F., et al. Qiskit: An open-source framework for quantum computing. *Accessed on: Mar*, 16, 2019.

Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J. C., Barends, R., Biswas, R., Boixo, S., Brandao, F. G., Buell, D. A., et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

Bang, J. and Yoo, S. A genetic-algorithm-based method to find unitary transformations for any desired quantum computation and application to a one-bit oracle decision problem. *Journal of the Korean Physical Society*, 65(12): 2001–2008, 2014.

Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A., and Weinfurter, H. Elementary gates for quantum computation. *Physical review A*, 52(5):3457, 1995.

Barkoutsos, P. K., Nannicini, G., Robert, A., Tavernelli, I., and Woerner, S. Improving variational quantum optimization using cvar. *Quantum*, 4:256, 2020.

Bartlett, R. J., Kucharski, S. A., and Noga, J. Alternative coupled-cluster ansätze ii. the unitary coupled-cluster method. *Chemical physics letters*, 155(1):133–140, 1989.

Bengio, Y., Leonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

Chang, J., Zhang, X., Guo, Y., Meng, G., Xiang, S., and Pan, C. Differentiable architecture search with ensemble gumbel-softmax. *CoRR*, abs/1905.01786, 2019. URL http://arxiv.org/abs/1905.01786.

Chu, X., Wang, X., Zhang, B., Lu, S., Wei, X., and Yan, J. Darts-: Robustly stepping out of performance collapse without indicators. In *International Conference on Learning Representation*, 2021.

Cong, I., Choi, S., and Lukin, M. D. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.

Deibuk, V. G. and Biloshytskyi, A. V. Design of a ternary reversible/quantum adder using genetic algorithm. *International Journal of Information Technology and Computer Science*, 7(9):38–45, 2015.

Dong, X. and Yang, Y. Searching for a robust neural architecture in four gpu hours. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1761–1770, 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Du, Y., Huang, T., You, S., Hsieh, M.-H., and Tao, D. Quantum circuit architecture search for variational quantum algorithms. *npj Quantum Information*, 8(1):1–8, 2022.

Elsken, T., Metzen, J., and Hutter, F. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 2019.

Erdős, P., Rényi, A., et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

Farhi, E., Goldstone, J., and Gutmann, S. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

Grimsley, H. R., Economou, S. E., Barnes, E., and Mayhall, N. J. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, 10(1):1–9, 2019.

Gumbel, E. Statistical theory of extreme values and some practical applications. *NBS Applied Mathematics Series*, 1954.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Herrmann, C., Bowen, R., and Zabih, R. Channel selection using gumbel softmax. In *European Conference on Computer Vision*, 2020.

Hoffmann, M. R. and Simons, J. A unitary multiconfigurational coupled-cluster method: Theory and applications. *The Journal of chemical physics*, 88(2):993–1002, 1988.

Hur, T., Kim, L., and Park, D. K. Quantum convolutional neural network for classical data classification. *Quantum Machine Intelligence*, 4(1):1–18, 2022.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

Jordan, P. and Wigner, E. P. Über das paulische äquivalenzverbot. In *The Collected Works of Eugene Paul Wigner*, pp. 109–129. Springer, 1993.

Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J. M., and Gambetta, J. M. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549(7671):242–246, 2017.

Kandasamy, K., Neiswanger, W., Schneider, J., Poczos, B., and Xing, E. P. Neural architecture search with bayesian optimisation and optimal transport. *Advances in neural information processing systems*, 31, 2018.

Karp, R. M. Reducibility among combinatorial problems. In *Complexity of computer computations*, pp. 85–103. Springer, 1972.

Kuo, E.-J., Fang, Y.-L. L., and Chen, S. Y.-C. Quantum architecture search via deep reinforcement learning. *arXiv preprint arXiv:2104.07715*, 2021.

Lamata, L., Alvarez-Rodriguez, U., Martín-Guerrero, J. D., Sanz, M., and Solano, E. Quantum autoencoders via quantum adders with genetic algorithms. *Quantum Science and Technology*, 4(1):014007, 2018.

Las Heras, U., Alvarez-Rodriguez, U., Solano, E., and Sanz, M. Genetic algorithms for digital quantum simulations. *Physical review letters*, 116(23):230504, 2016.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Li, R., Alvarez-Rodriguez, U., Lamata, L., and Solano, E. Approximate quantum adders with genetic algorithms: an ibm quantum experience. *Quantum Measurements and Quantum Metrology*, 4(1):1–7, 2017.

Liu, H., Simonyan, K., and Yang, Y. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Luo, J.H., W. J. L. W. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*, 2017.

Massey, P., Clark, J. A., and Stepney, S. Evolving quantum circuits and programs through genetic programming. In *Genetic and Evolutionary Computation Conference*, pp. 569–580. Springer, 2004.

McClean, J. R., Rubin, N. C., Sung, K. J., Kivlichan, I. D., Bonet-Monroig, X., Cao, Y., Dai, C., Fried, E. S., Gidney, C., Gimby, B., et al. Openfermion: the electronic structure package for quantum computers. *Quantum Science and Technology*, 5(3):034014, 2020.

Nautrup, H. P., Delfosse, N., Dunjko, V., Briegel, H. J., and Friis, N. Optimizing quantum error correction codes with reinforcement learning. *Quantum*, 3:215, 2019.

Nielsen, M. A. and Chuang, I. Quantum computation and quantum information, 2002.

Ostaszewski, M., Trenkwalder, L. M., Masarczyk, W., Scerri, E., and Dunjko, V. Reinforcement learning for optimization of variational quantum circuit architectures. *Advances in Neural Information Processing Systems*, 34: 18182–18194, 2021.

Potoček, V., Reynolds, A. P., Fedrizzi, A., and Corne, D. W. Multi-objective evolutionary algorithms for quantum circuit discovery. *arXiv preprint arXiv:1812.04458*, 2018.

Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pp. 4780–4789, 2019.

Richard, C. and David, H. *Methoden der mathematischen Physik*. Springer Berlin, Heidelberg, 1993.

Rigby, A. *Heuristics in quantum error correction*. PhD thesis, University of Tasmania, 2021.

Romero, J., Babbush, R., McClean, J., Hempel, C., Love, P., and Aspuru-Guzik, A. Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *arXiv preprint arXiv:1701.02691*, 2017.

Seeley, J. T., Richard, M. J., and Love, P. J. The bravyi-kitaev transformation for quantum computation of electronic structure. *The Journal of chemical physics*, 137 (22):224109, 2012.

Shor, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134. Ieee, 1994.

Slater, J. C. A simplification of the hartree-fock method. *Physical review*, 81(3):385, 1951.

Tang, H. L., Shkolnikov, V., Barron, G. S., Grimsley, H. R., Mayhall, N. J., Barnes, E., and Economou, S. E. qubit-adapt-vqe: An adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor. *PRX Quantum*, 2(2):020310, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017.

Wang, H., Ding, Y., Gu, J., Lin, Y., Pan, D. Z., Chong, F. T., and Han, S. Quantumnas: Noise-adaptive search for robust quantum circuits. In *IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 692–708. IEEE, 2022a.

Wang, X., Xue, C., Yan, J., Yang, X., Hu, Y., and Sun, K. Mergenas: Merge operations into one for differentiable architecture search. In *International Joint Conferences on Artificial Intelligence*, 2020.

Wang, X., Guo, W., Su, J., Yang, X., and Yan, J. Zarts: On zero-order optimization for neural architecture search. In *Neural Information Processing Systems*, 2022b.

Wang, X., Lin, J., Zhao, J., Yang, X., and Yan, J. Eautodet: Efficient architecture search for object detection. In *European Conference on Computer Vision*, 2022c.

Williams, C. P. and Gray, A. G. Automated design of quantum circuits. In *NASA International Conference on Quantum Computing and Quantum Communications*, pp. 113–125. Springer, 1999.

Williams, C. P., Clearwater, S. H., et al. *Explorations in quantum computing*. Springer, 1998.

Wu, Y., Bao, W.-S., Cao, S., Chen, F., Chen, M.-C., Chen, X., Chung, T.-H., Deng, H., Du, Y., Fan, D., et al. Strong quantum computational advantage using a superconducting quantum processor. *Physical review letters*, 127(18): 180501, 2021.

Ye, E. and Chen, S. Y.-C. Quantum architecture search via continual reinforcement learning. *arXiv preprint arXiv:2112.05779*, 2021.

Zeh, H. D. On the interpretation of measurement in quantum theory. *Foundations of Physics*, 1(1):69–76, 1970.

Zhang, S.-X., Hsieh, C.-Y., Zhang, S., and Yao, H. Differentiable quantum architecture search. *Quantum Science and Technology*, 7(4):045023, 2022.

Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018.

A. Preliminaries

Similar to a bit in the classical computation, a quantum bit (qubit) is the fundamental concept in quantum computing. Though a bit must be either in states 0 or 1, a qubit can be in a superposition of states $|0\rangle$ and $|1\rangle$, i.e. $|\psi_0\rangle = a\,|0\rangle + b\,|1\rangle$. $|\cdot\rangle$ is the Dirac notation. The complex number a and b satisfy $|a|^2 + |b|^2 = 1$.

The state space in quantum computing is defined on the Hilbert space. A quantum system can be completely described by a unit state vector $|\psi\rangle \in \mathbb{C}^d$. If the system is represented by n qubits, d will be equal to 2^n.

A quantum circuit can be used to simulate the evolution of a quantum system. A quantum gate, as the basic component of a quantum circuit, is instantiation of a unitary operator or unitary matrix U. The evolution process from $|\psi\rangle$ to $|\psi'\rangle$ through it can be expressed as $|\psi'\rangle = U\,|\psi\rangle$.

Quantum measurements will cause the collapse of a quantum state. For example, the qubit $|\psi_0\rangle$ will be either in states $|0\rangle$ or $|1\rangle$ instead of a superposition of $|0\rangle$ and $|1\rangle$ when measured. The probability of being measured as $|0\rangle$ and $|1\rangle$ is $|a|^2$ and $|b|^2$, respectively. We refer to the textbook (Nielsen & Chuang, 2002) for more detailed information on quantum computing.

B. Complexity of Our Approach

First, we analyze the spatial complexity of our method. The candidate gate set $\mathcal{G}$ has K candidates, taking $\mathcal{O}(K)$ space. The rotation parameters are allocated for each location in the circuit so it takes $\mathcal{O}(mn)$ space, where m and n denotes the number of layers and qubits, respectively. The architecture weights require $\mathcal{O}(K'K)$ memory for each location as explained in Sec.3.1. Hence, the memory cost for architecture weights is $\mathcal{O}(mnK'K)$. Then, the total memory cost is approximately $\mathcal{O}(mnK'K)$. When running the circuit, there exist unitary matrices with size $2^n \times 2^n$, which take additional $\mathcal{O}(4^n)$ space.

For time complexity, the bottleneck of our method is the forward process. For each iteration, it has to execute tensor product operations and matrix multiplication. The time complexity of tensor product operations in one layer can be calculated by:

$$2^2 \times 2^2 + 2^4 \times 2^2 + 2^6 \times 2^2 + \cdots + 2^{2(n-1)} \times 2^2 = 2^2 \times 2^2 \times \frac{2^{2(n-1)} - 1}{2^2 - 1} = \frac{4^{(n+1)} - 16}{3}.$$

Hence, tensor product operations take altogether $\mathcal{O}(4^n m)$ time. Then, matrix multiplication of a $2^n \times 2^n$ matrix and a $2^n \times 1$ matrix will be repeated m times in each iteration. Every multiplication takes $\mathcal{O}(4^n)$ time, adding up to total $\mathcal{O}(4^n m)$ time complexity. Supposing that the number of iterations is d, the overall time complexity of our algorithm will be $\mathcal{O}(4^n md)$.

Note that we utilize the basic tensor product operations and matrix multiplication to simulate the quantum circuit on classical computers, without leveraging any optimization skills. If equipped with such skills, the complexity of our algorithm will be further reduced.

C. Expressivity of Our Candidate Gate Set $\mathcal{G}$

Lemma C.1. *Any single-qubit quantum gate U can be decomposed into a sequence of R_z, R_y and R_z gates, and a phase (Barenco et al., 1995).*

$$U = e^{i\alpha} R_z(\theta_2) R_y(\theta_1) R_z(\theta_0)$$

Proof. The matrices of all the quantum gates are unitary. Hence, U can be rewritten as

$$U = e^{i\alpha} \begin{bmatrix} a & -b^* \\ b & a^* \end{bmatrix} = e^{i\alpha} V,$$

where a, b are complex while α is real, and $\det V = aa^* + bb^* = |a|^2 + |b|^2 = 1$ (* denotes the conjugate operator). Then, we have

$$\det U = e^{2i\alpha} \det V = e^{2i\alpha}.$$

Hence, we can obtain the phase angle α:

$$\alpha = \frac{1}{2} arctan2\big(Im(\det U), Re(\det U)\big).$$

Then, we plug in the matrices of rotation gates:

$$\begin{aligned}
U =& e^{i\alpha}\begin{bmatrix} e^{-i\frac{\theta_2}{2}} & 0 \\ 0 & e^{i\frac{\theta_2}{2}} \end{bmatrix}\begin{bmatrix} \cos\frac{\theta_1}{2} & -\sin\frac{\theta_1}{2} \\ \sin\frac{\theta_1}{2} & \cos\frac{\theta_1}{2} \end{bmatrix}\begin{bmatrix} e^{-i\frac{\theta_0}{2}} & 0 \\ 0 & e^{i\frac{\theta_0}{2}} \end{bmatrix} \\
=& e^{i\alpha}\begin{bmatrix} e^{-i\frac{\theta_0+\theta_2}{2}}\cos\frac{\theta_1}{2} & -e^{i\frac{\theta_0-\theta_2}{2}}\sin\frac{\theta_1}{2} \\ e^{i\frac{\theta_2-\theta_0}{2}}\sin\frac{\theta_1}{2} & e^{i\frac{\theta_0+\theta_2}{2}}\cos\frac{\theta_1}{2} \end{bmatrix} \\
=& e^{i\alpha}\begin{bmatrix} V_{00} & V_{01} \\ V_{10} & V_{11} \end{bmatrix}
\end{aligned}$$

Hence, we derive the angles of rotation gates:

$$\begin{aligned}
\theta_1 &= 2\arccos|V_{00}| \\
\theta_0+\theta_2 &= 2arctan2\big(Im(|V_{11}|), Re(|V_{11}|)\big) \\
\theta_2-\theta_0 &= 2arctan2\big(Im(|V_{10}|), Re(|V_{10}|)\big) \\
\theta_2 &= arctan2\big(Im(|V_{11}|), Re(|V_{11}|)\big) + arctan2\big(Im(|V_{10}|), Re(|V_{10}|)\big) \\
\theta_0 &= arctan2\big(Im(|V_{11}|), Re(|V_{11}|)\big) - arctan2\big(Im(|V_{10}|), Re(|V_{10}|)\big)
\end{aligned} .$$

Therefore, U is decomposed into a sequence of $\mathrm{R_z}$, $\mathrm{R_y}$ and $\mathrm{R_z}$ gates, and a phase. □

Corollary C.2. *Any single-qubit quantum gate U can be decomposed into a set of $\mathrm{R_zR_yR_z}$ and phase shift gates.*

Proof. Based on Lemma C.1, we just need to decompose $e^{i\alpha}$ into a set of $\mathrm{R_zR_yR_z}$ and the phase shift gates P:

$$\begin{aligned}
U =& e^{i\alpha}V \\
=& \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{i\alpha} \end{bmatrix} V \\
=& \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{-i\alpha} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & e^{2i\alpha} \end{bmatrix} V \\
=& R_z(-2\alpha)P(2\alpha)V \\
=& R_z(-2\alpha)R_y(0)R_z(0)P(2\alpha)V \\
=& R_z(-2\alpha)R_y(0)R_z(0)P(2\alpha)R_{\mathrm{z}}(\theta_2)R_{\mathrm{y}}(\theta_1)R_z(\theta_0)
\end{aligned} .$$

□

Theorem C.3. *The union of the set of single-qubit gates and CNOT is universal.* (*Williams et al., 1998*)

Since the $\mathrm{R_zR_yR_z}$ gate and the phase shift gate can express any single-qubit gate according to Corollary C.2, the set of $\mathrm{R_zR_yR_z}$, P and CNOT is universal based on Theorem C.3. Hence, the candidate gate set {$\mathrm{R_zR_yR_z}$, P, CNOT} can approximate any unitary matrix. We remove the phase shift gate P from our candidate gate set $\mathcal{G}$ because the global phase $e^{i\alpha}$ caused by P makes no difference to our results. Supposing the output state vector is $|\psi\rangle$, we add a global phase $e^{i\alpha}$ to it and get $|\psi_p\rangle = e^{i\alpha}|\psi\rangle$. For VQE tasks, the expectation value of the Hamiltonian $\mathcal{H}$ satisfies:

$$\langle\psi_p|\mathcal{H}|\psi_p\rangle = e^{-i\alpha}\langle\psi|\mathcal{H}|\psi\rangle e^{i\alpha} = \langle\psi|\mathcal{H}|\psi\rangle .$$

For image classification, the probability of the i-th basis state is calculated by $|\,|\psi\rangle^{(i)}\,|^2$, which satisfies:

$$|\,|\psi\rangle^{(i)}\,|^2 = |e^{i\alpha}|^2|\,|\psi\rangle^{(i)}\,|^2 = |e^{i\alpha}\,|\psi\rangle^{(i)}\,|^2 = |\,|\psi_p\rangle^{(i)}\,|^2.$$

Hence, the predicted probability of each class will not change. Therefore, the global phase has no effect on our experiment results and our candidate gate set $\mathcal{G}$={$\mathrm{R_zR_yR_z}$, I, CNOT} is sufficient for the expressivity in our tasks (the identity gate I is added as a placeholder).

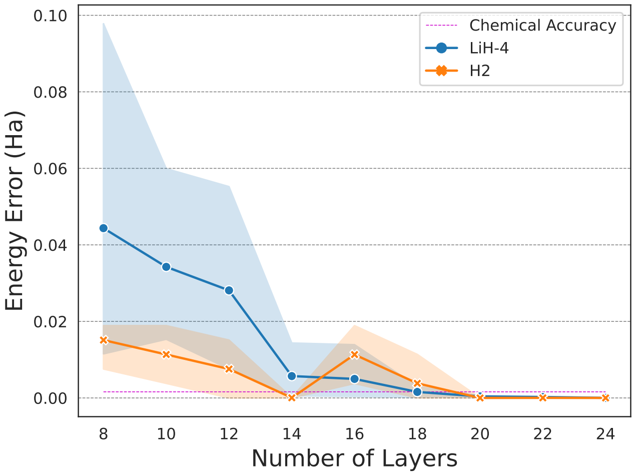


Figure 9. Relationship between the energy error and the number of layers. Every experiment is repeated five times. The dot or cross denotes the mean energy error of five trials, and the shade indicates the 95% confidence interval.

D. Supplements to the Ground State Energy Estimation Task

D.1. Molecule Information

We list the number of qubits, mapping methods and atom coordinates of each molecule in Tbl. 4. In particular, the Hamiltonians of LiH-4 and LiH-6 are taken directly from (Ostaszewski et al., 2021).

Table 4. Configurations of the molecules involved in our experiments. The coordinates are in angstrom (Å).

MOLECULE	#QUBIT	MAPPING	COORDINATES
H_2	4	JORDAN–WIGNER (JORDAN & WIGNER, 1993)	H (0, 0, -0.35) H (0, 0, 0.35)
LiH-4	4	PARITY (SEELEY ET AL., 2012)	Li (0, 0, 0) H (0, 0, 2.2)
LiH-6	6	JORDAN–WIGNER	Li (0, 0, 0) H (0, 0, 2.2)
H_2O-8	8	JORDAN–WIGNER	H (-0.021, -0.002, 0) O (0.835, 0.452, 0) H (1.477, -0.273, 0)

D.2. Sensitivity Analysis

Fig. 9 reveals the relation between the energy error and the number of layers. With the more layers, the mean energy error of both LiH-4 and H_2 quickly declines and attains the chemical accuracy when the number of layers reaches 20. In general, the performance benefits from the growth of the number of layers, and it will saturate at a certain point.

E. Supplements to the Unweighted Max-Cut Task

To derive the ultimate result we want, we need to decode the circuit output $|\psi\rangle$. We already know that $\langle\psi|\psi\rangle = 1$. Therefore, the probability $\vec{\tau}$ of basis states can be calculated by:

$$\vec{\tau} = |\psi\rangle \odot \overline{|\psi\rangle},$$

where $\odot$ means the hadamard product, and $\overline{|\psi\rangle}$ denotes the conjugate of $|\psi\rangle$. Take a 3-qubit circuit for example. Supposing that $\vec{\tau} = (0.0, 0.1, 0.1, 0.1, 0.3, 0.1, 0.1, 0.2)$, the probability of basis states $|000\rangle$, $|001\rangle$, $|010\rangle$, $|011\rangle$, $|100\rangle$, $|101\rangle$, $|110\rangle$ and $|111\rangle$ will be 0.0, 0.1, 0.1, 0.1, 0.3, 0.1, 0.1 and 0.2, respectively.

For unweighted Max-Cut, since the probability of basis state $|100\rangle$ is the biggest, the solution will be $x_1 = 1$, $x_2 = -1$, $x_3 = -1$ according to Eq. (8).

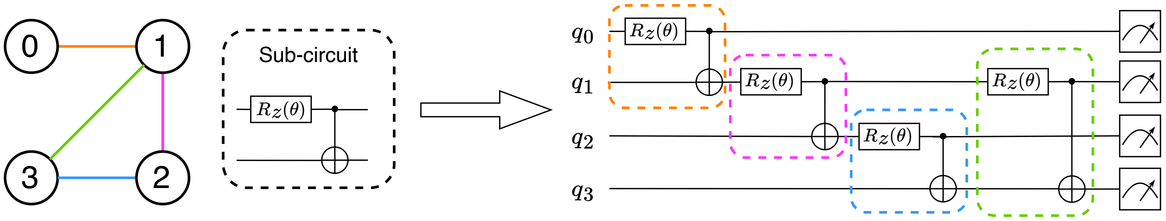


Figure 10. A detailed example of micro search for unweighted Max-Cut (best viewed in color). Each edge is matched with a sub-circuit. For instance, the green edge linking node 1 with node 3 is mapped to the green sub-circuit which acts on qubit 1 and qubit 3.

F. Supplements to the Image Classification Task

Autoencoder. Autoencoder is a neural network that first reduces the data to a compact encoding with an encoder, and then expands it back to the original dimension with a decoder. We use it to implement dimension reduction in our experiment. Following (Hur et al., 2022), the encoder and decoder are a fully connected layer with ReLU and sigmoid activation function, respectively. The hidden dimension is set as 8 for angle encoding and 16 for dense encoding. We train the autoencoder for 10 epochs on the training set with the mean squared error (MSE) loss. The Adam optimizer is used for optimization.

Data encoding schemes. To encode the data input $x_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(m)})$ into our quantum circuit (m is even), the angle encoding and dense angle encoding are leveraged because they require fewer gates to implement. Angle encoding maps the initial state $|\psi_0\rangle$ to

$$|\psi(x_i)\rangle = \bigotimes_{j=1}^{m} \Big(R_y(x_i^{(j)}) \, |0\rangle \Big) ,$$

while dense angle encoding maps it to

$$|\psi(x_i)\rangle = \bigotimes_{j=1}^{m/2} \Big(R_y(x_i^{(j+m/2)}) R_x(x_i^{(j)}) \, |0\rangle \Big) .$$

Fig. 11 shows our data encoding layers. Rotation gates are used to embed the reduced data input into our circuit.

Derivation of probability. To obtain the final result, we compute the probability $P\Big(\mathcal{M}\big(\,|\psi_t(x_i)\rangle\,\big) = |0\rangle\Big)$ by summing over the probability of even basis states, namely $|000\rangle$, $|010\rangle$, $|100\rangle$ and $|110\rangle$ (see the example in Appendix E). The probability $P\Big(\mathcal{M}\big(\,|\psi_t(x_i)\rangle\,\big) = |1\rangle\Big)$ is naturally the sum of odd basis states, i.e. $|001\rangle$, $|011\rangle$, $|101\rangle$ and $|111\rangle$. Obviously, we have $P\Big(\mathcal{M}\big(\,|\psi_t(x_i)\rangle\,\big) = |0\rangle\Big) + P\Big(\mathcal{M}\big(\,|\psi_t(x_i)\rangle\,\big) = |1\rangle\Big) = 1$.

G. Experimental Results for Other QAS Algorithms

Since there is no unified experimental settings, we cannot replicate the performance of some QAS approaches under our setting. Therefore, we list all the reported results and corresponding setting of the three tasks in this section to get a brief view of the performance of these QAS approaches.

G.1. Properties of Different QAS Algorithms

We list the properties of QCAS (Du et al., 2022), DQAS (Zhang et al., 2022), QuantumNAS (Wang et al., 2022a), RLQAS (Ostaszewski et al., 2021) and our model in Tbl. 5. The main difference between all the QAS methods is the architecture updating strategy. Sampling-based strategy (i.e. QCAS, DQAS, and QuantumNAS) is severely affected by the scaling problem. To achieve satisfactory results, the sampling time will become prohibitively long when the number of qubits increases. Updating through the reward function can achieve relatively good results on a certain problem, but the

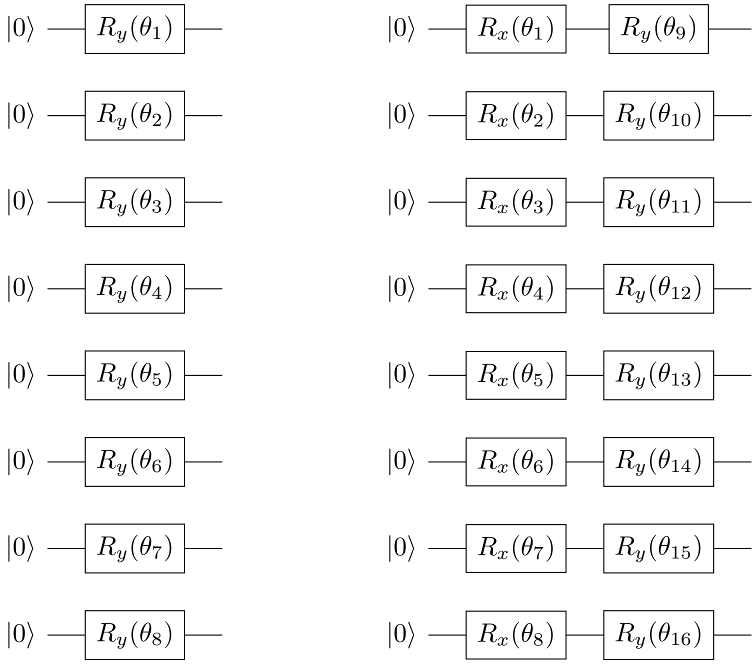


Figure 11. The structure of the angle encoding layer (left) and dense encoding layer (right). $\theta_.$ is derived from either PCA or the autoencoder and is scaled to $[0, \pi]$.

effort of finding the optimal reward function on a certain problem is comparable with the effort to manually design a circuit. Different types of noise are used and the error rate varies. The super-circuit stands for the pre-defined circuit structures to assist the searching methods. In our opinion, the super-circuits in QCAS, DQAS, and QuantumNAS may lack scalability and utilize too much prior. End-to-end learning refers to training the QAS model by applying gradient-based learning to the model as a whole (both architecture weights updating and rotation parameters updating), which improves the training efficiency. Some of the methods are not GPU-friendly or even do not support GPU, which makes the training of these methods relatively slow.

Table 5. Properties of different QAS methods. Self-defined* denotes 2% to 20% bit flip in between two gates.

Method	Architecture update	Noise	Super-circuit free	End-to-end	GPU support
QCAS	Sampling	Depolarizing	✗	✗	✗
DQAS	Monte-Carlo Sampling	Self-defined*	✗	✓	✗
QuantumNAS	Sampling	IBM-Q	✗	✗	✓
RLQAS	Reward Function	—	✓	✗	✓
Ours	Gradient Descent	Readout	✓	✓	✓

G.2. Ground State Energy Estimation

We find other two QAS approaches who have listed ground state energy estimation as one of their experiments. (Ostaszewski et al., 2021) reports numerical results within the chemical accuracy, while (Wang et al., 2022a) tests their algorithm under simulated circuit noise, which strongly affect the estimation accuracy.

(Ostaszewski et al., 2021) uses RL to search for VQE ansätze. They propose a reward policy based on the energy of a given Hamiltonian. The chemical accuracy is used as a threshold, forming the condition for final reward together with maximum layers. The gates are encoded into 4 decimal numbers, which indicate CNOT control qubit, CNOT target qubit, rotation qubit and rotation axis. The authors evaluate their algorithm on two settings. The first experiment is conducted with LiH-4bit Hamiltonian with three values of bond distances, 1.2Å, 2.2Å and 3.4Å. The second experiment is conducted on the Jordan-Wigner mapping with the LiH-6bit Hamiltonian with the bond distance at 2.2Å. For each Hamiltonian, the authors give ten trials and count the number of trials achieving the chemical accuracy. In Experiment 1, the authors find that using Global COBYLA optimization can achieve the best results. In Experiment 2, the proposed RL algorithm achieves the chemical accuracy only in 7 out of 10 trials, and the minimum energy gap ever achieved during the training process is slightly less than 1×10^{-3} (no exact number in the paper), where our algorithm achieves 2.94×10^{-4}. Our algorithm outperforms the results of (Ostaszewski et al., 2021) from both accuracy and stability. However, (Ostaszewski et al., 2021) report that they achieve the results with very few gates as well as very low circuit depth. They only use an average of 16 layers and 35 gates to achieve the results for LiH 6-qubit Hamiltonian. This indicates that the circuit only have an average of 2 gates on each layer, which is totally different from our results. To sum up, our algorithm achieves better performance compared to this RL approach with more gates (which is still much less than what UCCSD needs). Furthermore, we also test our algorithm with more complicated Hamiltonians and different molecules.

(Wang et al., 2022a) mainly introduces the TorchQuantum library proposed by them and focuses on designing noise adaptive quantum circuit for VQE algorithms. The proposed algorithm is a sampling based algorithm. They first construct and train the SuperCircuit and then apply a noise-adaptive evolutionary co-search with iterative quantum pruning. The authors report intensive results on LiH-6bit, H_2O-6bit, CH_4-6bit, CH_4-10bit and BeH_2-15bit. However, with the influence of the circuit noise, the gap between the optimal energy and the obtained energy is rounded to 1 decimal place, and the gap for UCCSD is more than 10 on H_2O-6bit, CH_4-6bit and CH_4-10bit. The circuits obtained from the algorithm have much fewer gates compared to UCCSD and trained under the noise environment, which explain why their performance is better than UCCSD under the noisy environment. However, 100 times the chemical accuracy is no better than 10000 times the chemical accuracy. Thus, we do not test our algorithm under the NISQ device and we will wait for the quantum error correction code to lead quantum computing from the NISQ era to the next noiseless era.

G.3. Unweighted Max-Cut

Another QAS method reporting having successfully solved the Unweighted Max-Cut problem is (Zhang et al., 2022), which is also to our knowledge, the only published work as a differentiable searching algorithm with both architecture weights and rotation parameters updated by gradient descent. The authors use Monte Carlo sampling to approximate the gradient. Comparing to our method, which uses Gumbel-Softmax to sample one circuit at each epoch, (Zhang et al., 2022) still needs to sample multiple circuits at each epoch to approximate the continuous distribution of the weights. The experimental setting of this paper involves additional uniquely defined gates for QAOA in the candidate set and also associates the connectivity between nodes with the group of gates connecting qubits. The experiments are conducted under noiseless environment and only have 8-node graphs for Max-Cut. No circuit depth or number of gates or running time is reported. Note that this paper leverages the CVaR metric (Barkoutsos et al., 2020), which equals the average cut value of the top 20% cut values in all measurements, to evaluate their method in Max-Cut. Hence, we also apply this metric to our model. The CVaR results for graphs in Fig.5 are 13, 15 and 24, respectively. All of them are equal to the maximal cut values while (Zhang et al., 2022) fails to reach the maximal cut value in its graph instance.

G.4. Image Classification

(Wang et al., 2022a) also reports results on classification tasks. Similar to ground state energy estimation, (Wang et al., 2022a) conducts large amounts of experiments under very complicated settings. No comparison against classical CNN or QCNN is reported and the results are also severely affected by the circuit noise. Since the paper mainly focuses on noise-adaptive circuit search and the setting is also too complicated, we fail to follow the experiment setting of this paper. The results for 2-class image classification for number 3 and 6 is around 0.95 accuracy on IBMQ-Yorktown, which is the closest setting we can find to us (more than 0.99 accuracy on 0-1 classification with PCA encoding).

H. Ablation Study

We implement the decomposition of the architecture parameter $\mathbf{A_{ij}}$ through two sequential linear layers. Tbl. 6 shows the influence of the decomposition module. The experiment is performed under the same setup as Sec. 4.3.2. With the decomposition module, the accuracy of our model increases by nearly 5 percent. Therefore, this module enhances the expressivity of our architecture parameters, while only a trivial computation overhead is added.

Table 6. The influence of the decomposition module. Each experiment is repeated five times.

MODEL	PCA-ANGLE	AE-ANGLE	PCA-DENSE	AE-DENSE
OURS	0.9920 ± 0.0011	0.9809 ± 0.0119	0.9922 ± 0.0018	0.9742 ± 0.0116
OURS WITHOUT DECOMPOSITION	0.9764 ± 0.0200	0.9410 ± 0.0487	0.9167 ± 0.0455	0.9252 ± 0.0296

I. Example Circuits Obtained by Our Algorithm

We visualize some circuits found by our model with Qiskit as below.

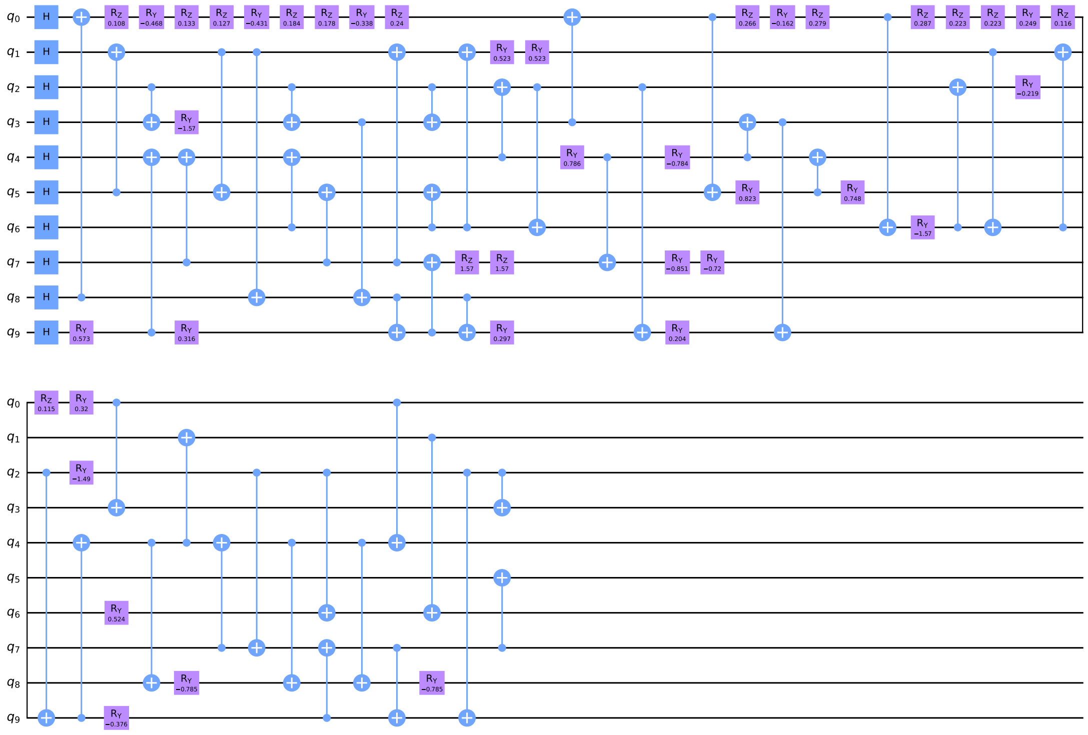


Figure 12. The generated circuit for the graph in Fig. 5a. The number of parameters, number of gates, and depth are 41, 99 and 30.

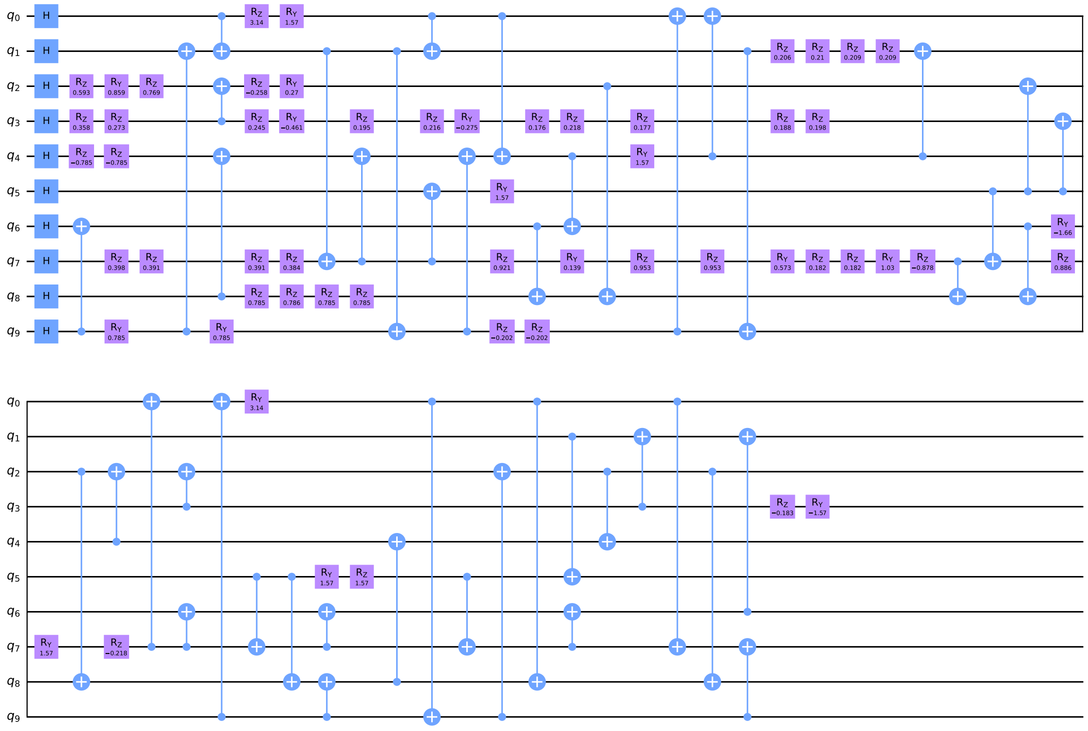


Figure 13. The generated circuit for the graph in Fig. 5b. The number of parameters, number of gates, and depth are 57, 114 and 33.

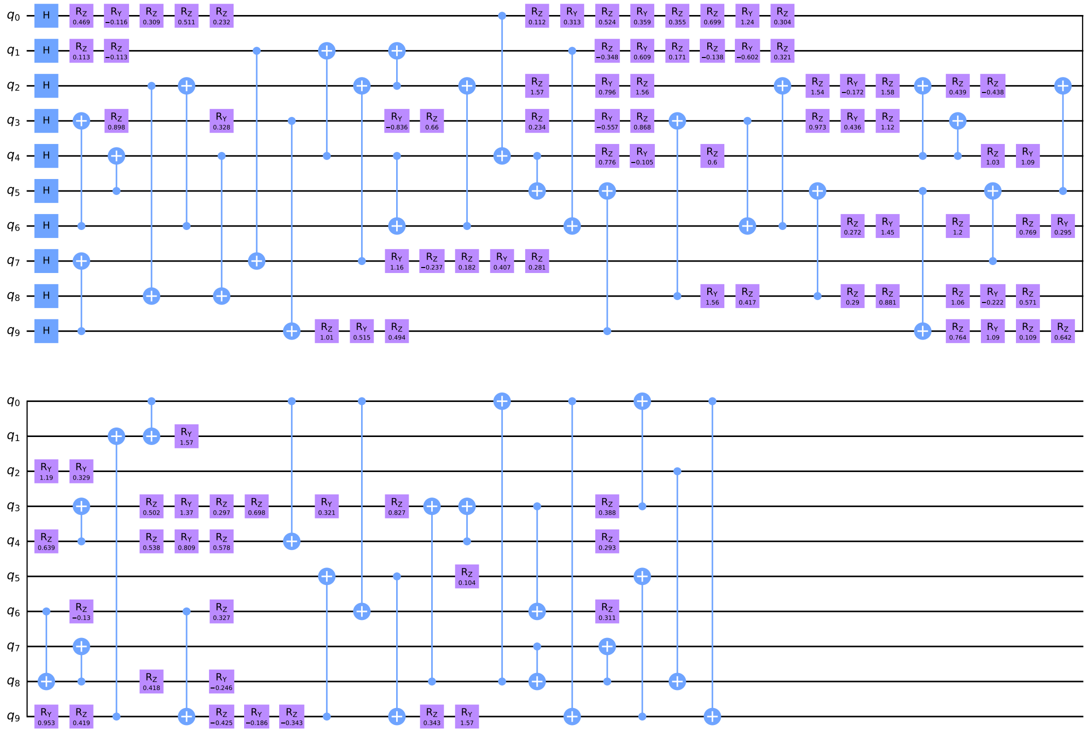


Figure 14. The generated circuit for the graph in Fig. 5c. The number of parameters, number of gates, and depth are 96, 153 and 34.