MG-Net: Learn to Customize QAOA with Circuit Depth Awareness

Yang Qian[1,†] **Xinbiao Wang**[2,‡] **Yuxuan Du**[3,§ *] **Yong Luo**[2,¶] **Dacheng Tao**[3,◇]
[1]School of Computer Science, Faculty of Engineering, University of Sydney
New South Wales 2008, Australia
[2]Institute of Artificial Intelligence, School of Computer Science, Wuhan University
Wuhan, China
[3]College of Computing and Data Science, Nanyang Technological University
Singapore 639798, Singapore
[†]`qianyang1217@gmail.com` [‡]`cyriewang@gmail.com` [§]`duyuxuan123@gmail.com`
[¶]`luoyong@whu.edu.cn` [◇]`dacheng.tao@ntu.edu.sg`

Abstract

Quantum Approximate Optimization Algorithm (QAOA) and its variants exhibit immense potential in tackling combinatorial optimization challenges. However, their practical realization confronts a dilemma: the requisite circuit depth for satisfactory performance is problem-specific and often exceeds the maximum capability of current quantum devices. To address this dilemma, here we first analyze the convergence behavior of QAOA, uncovering the origins of this dilemma and elucidating the intricate relationship between the employed mixer Hamiltonian, the specific problem at hand, and the permissible maximum circuit depth. Harnessing this understanding, we introduce the Mixer Generator Network (MG-Net), a unified deep learning framework adept at dynamically formulating optimal mixer Hamiltonians tailored to distinct tasks and circuit depths. Systematic simulations, encompassing Ising models and weighted Max-Cut instances with up to 64 qubits, substantiate our theoretical findings, highlighting MG-Net's superior performance in terms of both approximation ratio and efficiency.

1 Introduction

Combinatorial optimization problems (COPs) [1], central to numerous scientific and engineering disciplines [2, 3, 4], often defy efficient classical solutions due to their computational complexity [5, 6]. A promising strategy to overcome these computational challenges involves harnessing the power of quantum computing, as these COPs can be mapped to Ising Hamiltonians whose ground states denote optimal solutions [7, 8]. Leveraging this quantum representation, the Quantum Approximate Optimization Algorithm (QAOA) [9] has emerged to address these COPs. In particular, theoretical analyses [10, 11, 12, 13] underscore the potential of QAOA, suggesting its superiority over classical counterparts in certain contexts, particularly with unlimited infinite circuit depth. Meantime, empirical studies [14, 15, 16] affirm its applicability across a diverse spectrum of problems and devices.

Despite these advancements, QAOA's practical efficacy is challenged by the quantum coherence limits of modern quantum devices, as there is a ceiling on the allowable maximum circuit depth p. As a result, standard QAOA often underperforms classical counterparts [18, 19]. This motivates a research shift towards redesigning the *mixer Hamiltonian* H_M, a key component of QAOA. As illustrated in Fig. 1(a), supported by the results of quantum adiabatic evolution [20, 21], alternative

[*]Corresponding authors

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-1061

H_M may exist that guide the system along a more direct and efficient trajectory—a shortcut—to the solution state, leading to a better performance compared to the standard QAOA. Besides, as shown in Fig. 1(b), empirical evidence indicates that the form of H_M promising a good performance is varied with the allowable p. As such, diverse alternatives H_M are proposed in past years, drawing upon concepts from quantum annealing [22], incorporating additional trainable parameters [17] or exploiting permutation symmetry [23]. However, these approaches require deep domain expertise and often lack generalizability across different tasks and circuit configurations p.

In response to these challenges, here we first analyze the convergence of QAOA on various mixer Hamiltonian configurations and circuit depths with the tool of representation theory [24]. Our finding reveals that (i) the convergence of QAOA can be enhanced through parameter grouping in the mixer Hamiltonian; (ii) the specific strategy for parameter grouping is dependent on the particular problem and the value of p. These two findings are instrumental in understanding the interplay between p, parameter grouping, and the overall efficiency of the QAOA, providing valuable insights for the design of the mixer Hamiltonian.

Envisioned by the achieved theoretical results, we propose an end-to-end learning framework, termed **M**ixer **G**enerator **Net**work (MG-Net), to dynamically design the mixer Hamiltonian H_M for a class of problems and distinct circuit depth constraints. Conceptually, MG-Net takes the problem's description and the available circuit depth p as input and directly outputs the optimal mixer Hamiltonian for a p-QAOA. There are three distinguished features of our proposal: (i) The ability to dynamically adjust H_M according to p, enhancing its *compatibility* with practical quantum devices; (ii) Fast customization of H_M for *unseen problems* and circuit depth p, attributed to the multi-condition controlled generative network architecture; (iii) Circumvent the need for the expensive collection of a vast training dataset of optimal H_M by employing an estimator-generator structure alongside a two-stage training approach. Note that the developed techniques can be flexibly extended to other variational quantum algorithms (VQAs) [25, 26], which may be independent of interests.

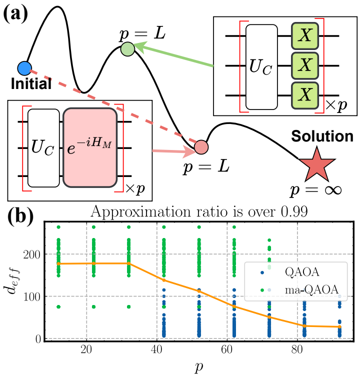

Figure 1: **Mixer Hamiltonian affects the performance of QAOA**. (a) The optimization trajectories of QAOA with varied mixer Hamiltonians H_M. Given a fixed circuit depth p, a tailored H_M (highlighted in pink) can more effectively steer the quantum state towards the exact solution compared to the original H_M used in QAOA. (b) Transition of the effective dimension d_{eff} required in QAOA with increasing p. 'ma-QAOA' denotes a case with independent parameters [17], contrasted with 'QAOA' where parameters are fully correlated. The orange line denotes the average effective dimension over all samples.

The contributions of this paper are:
• We provide a rigorous theoretical analysis on the convergence of QAOA with sufficient circuit depth, elucidating **the link between the performance and the parameter grouping in QAOA circuits**. This analysis offers guidance on the design of mixer Hamiltonian to achieve a high approximation ratio for a specified circuit depth.
• We propose MG-Net, which dynamically tailors its predicted mixer Hamiltonian H_M to suit the given problem and circuit depth. Our model **greatly reduces the cost of collecting labeled training data**, attributed to an estimator-generator framework and a two-stage training strategy.
• The proposed MG-Net demonstrates remarkable **generalization ability** from a limited dataset to a broad spectrum of combinatorial problems, which facilitates rapid and efficient creation of H_M for unseen problems, advancing the **practical utility** of QAOAs.
• Extensive experiments on the Transverse-field Ising model and Max-Cut up to 64 qubits **verify our theoretical discoveries** and **demonstrate the advantage of MG-Net in achieving higher approximation ratios at various circuit depths** compared to other quantum and traditional methods. The code is released at `https://github.com/QQQYang/MG-Net`.

2 Background

2.1 Quantum approximation optimization algorithm

Considering a COP defined on a set of N binary variables $\boldsymbol{z} = z_1 \cdots z_N$, where $z_i \in \{\pm 1\}$, our objective is to identify a bit string $\boldsymbol{z}$ that maximizes a specific objective function $C(\boldsymbol{z}) : \{\pm 1\}^N \to \mathbb{R}_{\geq 0}$. Intuitively, the solution space grows exponentially with N, rendering the exact solution to many COPs intractable [1]. In practice, an alternative approximation algorithm is selected to seek an approximate solution $\boldsymbol{z}$ to achieve a high approximation ratio $r = C(\boldsymbol{z})/C_{\max}$, where $C_{\max} = \max_{\boldsymbol{z}} C(\boldsymbol{z})$.

In response to this inherent complexity, Quantum Approximate Optimization Algorithm (QAOA) [9] is proposed. In this framework, the bit string $\boldsymbol{z}$ is encoded into a quantum state $|\boldsymbol{x}\rangle = |x_1 \cdots x_N\rangle$ with $x_i = (1 - z_i)/2$, and the objective function $C(\boldsymbol{x})$ is encoded into the problem Hamiltonian $H_C \in \mathbb{C}^{2^N \times 2^N}$ so that $H_C |\boldsymbol{x}\rangle = C(\boldsymbol{x}) |\boldsymbol{x}\rangle$. Refer to Appendix A for the omitted details.

QAOA is a hybrid quantum-classical algorithm that combines a parameterized quantum circuit (PQC) for state evolution and a classical optimizer for parameter updates. For a p-layer QAOA circuit shown in Fig. 1(a), the quantum state $|\psi_p\rangle$ is prepared by alternately applying the problem Hamiltonian H_C and the mixer Hamiltonian $H_M = \sum_{i=1}^{N} X_i$ on the initial state $|\psi_0\rangle$, formulated as

$$|\psi_p(\boldsymbol{\alpha}, \boldsymbol{\beta})\rangle = \prod_{k=1}^{p} e^{-i\beta_k H_M} e^{-i\alpha_k H_C} |\psi_0\rangle , \tag{1}$$

where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_p)$ and $\boldsymbol{\beta} = (\beta_1, ..., \beta_p)$ are $2p$ trainable parameters. These parameters are optimized to maximize the expectation value of the problem Hamiltonian H_C:

$$(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \arg\max_{\boldsymbol{\alpha}, \boldsymbol{\beta}} F_p(\boldsymbol{\alpha}, \boldsymbol{\beta}), \tag{2}$$

where $F_p(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \langle \psi_p(\boldsymbol{\alpha}, \boldsymbol{\beta}) | H_C | \psi_p(\boldsymbol{\alpha}, \boldsymbol{\beta}) \rangle$ can be estimated by multiple measurements on the quantum system. As $F_p(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ approaches the optimal value $C_{\max}$ of the objective function, we can obtain the approximate solution to the combinatorial optimization problem with high probability by measuring the state $|\psi_p(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)\rangle$ in the computational basis. A metric for assessing the performance of QAOA is the approximation ratio $r = F_p(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)/C_{\max}$.

2.2 Symmetry in QAOA

Symmetry, ansatz design, and effective dimension. A symmetry S refers to the unitary operator leaving the operator H invariant such that $S^\dagger H S = H$ (or $[S, H] = 0$). All symmetries form a group $\mathcal{S}$ where given any two symmetries $S_1, S_2 \in \mathcal{S}$, the compositions $S_1 S_2$ and $S_2 S_1$ are also symmetries in $\mathcal{S}$. Among various symmetries, the most relevant one to our work is the permutation symmetry $\pi \in \mathcal{S}_N$, with the subscript being the qubit count N and $\mathcal{S}_N$ being the symmetric group. For example, a permutation π with $\pi(1) = 3, \pi(2) = 1, \pi(3) = 2$ acting on the state $|\psi_1\rangle |\psi_2\rangle |\psi_3\rangle$ yields $\pi |\psi_1\rangle |\psi_2\rangle |\psi_3\rangle = |\psi_3\rangle |\psi_1\rangle |\psi_2\rangle$. Throughout the whole study, we denote the group of permutation symmetries of the problem Hamiltonian H_C as $\mathrm{Per}(H_C) = \{\pi \in \mathcal{S}_N \mid \pi^\dagger H_C \pi = H_C\}$.

Consider an N-qubit PQC $U(\boldsymbol{\theta}) = \prod_{j=1}^{p} \prod_{k=1}^{K} e^{-iH_k \boldsymbol{\theta}_{jk}}$ with $\boldsymbol{\theta} \in \Theta$ and $d = 2^N$. We call $U(\boldsymbol{\theta})$ a symmetric PQC with respect to the problem Hamiltonian H_C if there exists a symmetry group $\mathcal{S}$ of H_C such that $[U(\boldsymbol{\theta}), S] = 0$ for any $\boldsymbol{\theta} \in \Theta$ and $S \in \mathcal{S}$. This symmetry is determined by the generators of PQCs $\mathcal{A} = \{H_1, \cdots, H_K\}$ which is also called *ansatz design*, as $[U(\boldsymbol{\theta}), S] = 0$ holds for any $\boldsymbol{\theta} \in \Theta$ if and only if $[H_k, U(\boldsymbol{\theta})] = 0$ for any $k \in [K]$. Such symmetry can be quantified by the *effective dimension* [27, 28].

Definition 2.1 (Effective dimension)**.** Consider an N-qubit QAOA instance $(|\psi_0\rangle, U(\boldsymbol{\theta}), H_C)$ where $U(\boldsymbol{\theta})$ acts on the vector space V. If there exists a direct sum decomposition $V = \oplus_{j=1}^{k} V_j$ and $V^* \in \{V_j\}_{j=1}^{k}$ such that $U(\boldsymbol{\theta}) |\psi_0\rangle \in V^*$ for any $\boldsymbol{\theta}$ and the ground state of the problem Hamiltonian $|\psi^*\rangle$ satisfies $|\psi^*\rangle \in V^*$, then the effective dimension $d_{\text{eff}} \leq 2^N$ is defined as the dimension of V^*.

Experimental and theoretical analysis has shown that symmetric ansatz design with a small effective dimension contributes to better trainability [29, 28, 30].

https://doi.org/10.52202/079017-1061

Symmetry and ansatz designs in QAOA. The PQC in Eqn. (1), adopted in the original QAOA, fully groups (FG) the trainable parameters and has the ansatz design $\mathcal{A}_{FG} = \{H_M, H_C\}$, which is symmetric with respect to H_C under the *permutation symmetry*. This is because its mixer Hamiltonian $H_M = \sum_{i=1}^{N} X_i$ is invariant under an arbitrary permutation operator.

However, $\mathcal{A}_{FG}$ fails to employ the specific symmetry group of H_C. This issue can be addressed by partially grouping (PG) trainable parameters in QAOA. For example, denote $H_C = \sum_{(i_k, j_k)} Z_{i_k} Z_{j_k}$ with Z_{j_k} being the Pauli-Z operator acting on the j_k-th qubit. An alternative symmetric ansatz design is $\mathcal{A}_{PG} = \{H_{\mathcal{O}_1}, \cdots, H_{\mathcal{O}_{|\mathcal{O}|}}, H_{\mathcal{O}^e_1}, \cdots, H_{\mathcal{O}^e_{|\mathcal{O}^e|}}\}$ where $H_{\mathcal{O}_k} = \sum_{i \in \mathcal{O}_k} X_i$ and $H_{\mathcal{O}^e_k} = \sum_{(i,j) \in \mathcal{O}^e_k} Z_i Z_j$ refer to the generators respecting the permutation symmetry of H_C satisfying $H_M = \sum_{j=1}^{|\mathcal{O}|} H_{\mathcal{O}_j}$ and $H_C = \sum_{j=1}^{|\mathcal{O}^e|} H_{\mathcal{O}^e_j}$ [23]. The ansatz design $\mathcal{A}_{PG}$ enables more free parameters than the ansatz design $\mathcal{A}_{FG}$ in each layer, and has been empirically shown with a faster convergence rate than $\mathcal{A}_{FG}$ given the same number of layers.

When H_C is asymmetric, another typical ansatz design in QAOA is $\mathcal{A}_{NG} = \{Z_{i_1} Z_{j_1}, \cdots, Z_{i_k} Z_{j_k}, X_1, \cdots, X_N\}$, where the parameters of all parameterized gates are independent and non-grouping (NG). Notably, the PQCs related to various ansatz design $\mathcal{A}_{FG}, \mathcal{A}_{PG}, \mathcal{A}_{NG}$ employ the same parameterized gates but with different *parameter grouping strategies*, where $(\boldsymbol{\beta}, \boldsymbol{\alpha})$ in each layer can be fully grouped, partially grouped, and non-grouped [23].

3 Convergence theory of QAOA

In this section, we theoretically illustrate how employing appropriate parameter grouping corresponds to better convergence performance. Similar to Refs. [27] and [28], our derivations are based on the observation that the exploited PQC with highly-symmetric ansatz structure generally enables a faster convergence rate.

Theorem 3.1 (Convergence)**.** *Consider a QAOA instance denoted as* $(|\psi_0\rangle, U(\boldsymbol{\theta}), H_C)$ *with* $U(\boldsymbol{\theta})$ *determined by the related ansatz design. Let* $\mathcal{A}_{FG}, \mathcal{A}_{PG}, \mathcal{A}_{NG}$ *be the ansatz designs of the circuits with parameters fully grouped, partially grouped, and no-grouped. Their effective dimension yields*

$$d_{\text{eff}}(\mathcal{A}_{FG}) = d_{\text{eff}}(\mathcal{A}_{PG}) \leq d_{\text{eff}}(\mathcal{A}_{NG}), \tag{3}$$

where the equality in the inequality holds if there is no spatial symmetry in H_C*. Besides, there exists a* d_{eff}*-dependent threshold* C *so that circuit depth* $p > C$*, the iterations* T *required to achieve the same approximation ratio yield*

$$T_{PG} = T_{FG} \leq T_{NG}. \tag{4}$$

The proof of Theorem 3.1 and more elaborations are presented in Appendix B. The achieved results, combined with the over-parameterization theory of PQCs [30], deliver the following two implications. First, when the circuit depth $p > C$ is sufficiently large such that all PQCs with various ansatz designs reach the *over-parameterization regime*, performing the parameter grouping can effectively decrease the effective dimension d_{eff} compared with the PQCs with no-parameter grouping, leading to a faster convergence rate. Second, the over-parameterization of QAOA occurs when the number of trainable parameters exceeds a critical point that is proportionally related to d_{eff}.

The above two implications indicate the selection of $\mathcal{A}_{FG}$, $\mathcal{A}_{PG}$, or $\mathcal{A}_{NG}$ is complicated and is both *depth- and problem-dependent*. In particular, given a specified p, adopting a parameter grouping strategy can simultaneously reduce the number of parameters and the effective dimension, making it difficult to determine whether the QAOA reaches the over-parameterization regime. For instance, in a scenario such that the parameter grouping strategy drastically reduces the number of parameters but only slightly reduces the effective dimension, an over-parameterized QAOA could transform to an under-parameterized QAOA, leading to a degraded convergence as the optimization can be easily stuck in bad local minimal [31, 32].

4 MG-Net

The implication of Theorem 3.1 inspires us to devise a method for dynamically generating an appropriate mixer Hamiltonian H_M tailored to both the problem G at hand and the specified circuit depth p. For this purpose, we harness the power of deep learning and devise an end-to-end learning framework, dubbed Mixer Generator Network (MG-Net).

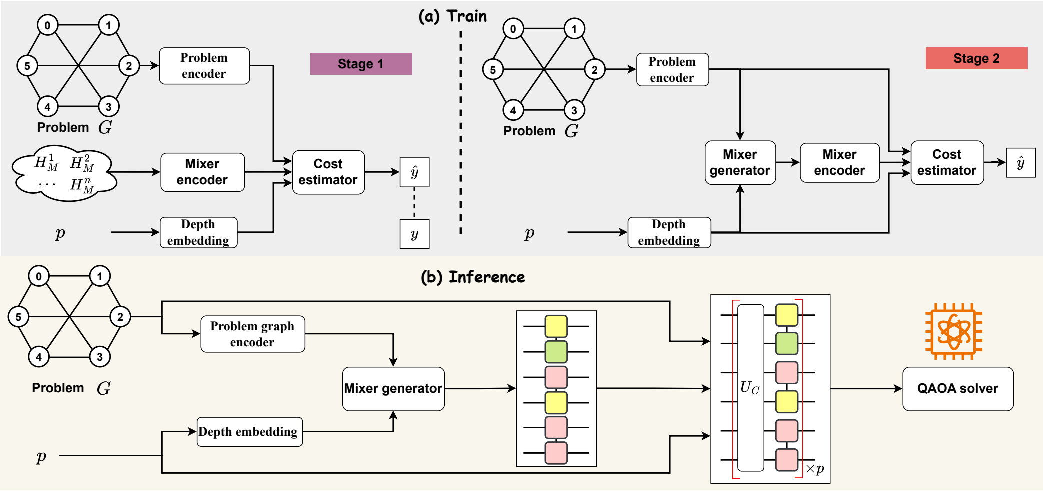

Figure 2: **Framework of MG-Net.** (a) Training Phase. Initially (left), the cost estimator is trained to precisely predict QAOA performance for specific problem instances, circuit depths, and mixer Hamiltonians. In the subsequent stage (right), with the cost estimator fixed, the mixer generator is trained through unsupervised learning to derive the optimal mixer Hamiltonian that minimizes the cost estimator's output. (b) Inference Phase. Given a problem G and circuit depth p, the mixer generator produces a mixer Hamiltonian, subsequently utilized in a QAOA solver to find the solution.

4.1 Framework of MG-Net

Before presenting the proposed MG-Net, let us first formalize the learning problem towards designing the mixer Hamiltonian H_M. To incorporate different Pauli operators and parameter grouping strategies, we extend the definition of an N-qubit mixer Hamiltonian H_M in Eqn. (1) to a more generalized form, supporting flexible operators and parameter correlations by substituting the Pauli-X operator with a selection of general Pauli operator and stratifying the N operators into K groups. Mathematically, the refined mixer Hamiltonian yields

$$H_M = \sum_{j=1}^{K} \beta_j \sum_{i \in \mathcal{G}_j} P_i, \tag{5}$$

where β_j refers to the trainable parameter controlling the j-th group of operators, $\mathcal{G}_j$ contains the indices of operators belonging to the j-th group such that $\cup_{j=1}^{K} \mathcal{G}_j = [N]$ and $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$ for $\forall i \neq j$, and P_i refers to a Pauli operator. This work focuses on $P_i \in X, Y$ as the types of candidate operators, rather than using only the single Pauli-X operator. Previous studies have demonstrated the effectiveness of using Pauli-Y operators in mixer Hamiltonians [33, 34], as summarized in Tab. 1. In this sense, operators in the same group are correlated with each other, sharing the same parameter. In this way, *the design of H_M is decoupled into two distinct tasks: determine the parameter groups $\{\mathcal{G}_j\}_{j=1}^{K}$; identify the appropriate operator types P_i*. With the reformulation above, the decoupled tasks can be accomplished by learning a mapping rule $f : (G, p) \rightarrow (\mathcal{G}, \mathcal{P})$ with $\mathcal{G} = \{\mathcal{G}_j\}_{j=1}^{K}$ and $\mathcal{P} \in \{X, Y\}^{\otimes N}$ referring to the parameter correlation and mixer Hamiltonian.

Table 1: **Previous works that have introduced the Pauli-Y operator as a candidate mixer Hamiltonian.**

Works	Mixer Hamiltonian
DC-QAOA [33]	$\{X, Y, ZY, YZ, XY, YX\}$
ADAPT-QAOA [34]	$\cup_{i\in[N]}\{X_i, Y_i\} \cup \{\sum_{i\in[N]} Y_i\} \cup \{\sum_{i\in[N]} X_i\} \cup_{i,j\in[N]\times[N]} \{B_iC_j \mid B_i, C_j \in \{X, Y, Z\}\}$

Designing a model to learn f faces *two main challenges*:
(C-1) The variety of combinatorial optimization tasks leads to uncertain input formats for the model, which necessitates a universal representation method and retains essential properties of the original

data, such as permutation invariance;
(C-2) The exponential growth of the search space for both parameter correlation and operator types, (i.e., scaling at $O(N^N)$ and $O(2^N)$, respectively), hurdles the design of an effective learning method. For instance, directing training a learning model in the supervised learning paradigm may require computationally unaffordable training examples to ensure good prediction accuracy.

We next present an end-to-end learning framework—**M**ixer **G**enerator **Net**work (MG-Net), as depicted in Fig. 2, to address the above challenges. Particularly, to address **C-1**, we devise a problem encoder which transforms each problem G into a unified directed acyclic graph G_C, ensuring a consistent and effective input format. Coupled with the mixer encoder, it maps both the problem and mixer Hamiltonian to a shared hidden space. To address **C-2**, MG-Net features a unique *estimator-generator* framework, supplemented by a *two-stage training strategy*. The role of these techniques is summarized below and their implementation details are demonstrated in the subsequent subsections.

Role of estimator. Rather than directly seeking the optimal parameter correlation strategy $\mathcal{G}^*$ and operator type $\mathcal{P}^*$ for a given (G, p), we devise a *cost estimator* to map the relationship between $(\mathcal{G}, \mathcal{P})$ and the achievable minimal cost F_p of the corresponding QAOA in Eqn. (2).

Role of generator. We devise a *generator* to predict $(\mathcal{G}, \mathcal{P})$ that minimizes the cost estimator's output. This design requires only the cost of any mixer Hamiltonian as a label, thus avoiding the exhaustive search of optimal pairs $(\mathcal{G}^*, \mathcal{P}^*)$.

Two-stage training. The pipeline is visualized in Fig. 2(a).
• **Stage 1 (Cost Estimator Training)**. This stage, marked in purple, focuses on training the cost estimator using *supervised learning*. Inputs include the problem graph G, potential mixer Hamiltonians H_M, and the chosen circuit depth p, with the corresponding cost y as the target label.
• **Stage 2 (Mixer Generator Training)**. This stage, marked in orange, freezes the cost estimator and only updates the mixer generator to minimize the output of the cost estimator under the *unsupervised learning* paradigm.

For inference on unknown problem instances (in Fig. 2(b)), MG-Net employs only the mixer generator to predict the optimal mixer Hamiltonian, which is then fed into a QAOA solver to derive the final solution. Distinguished by its ability to generalize effectively across a class of problems from a limited learning set, MG-Net sets itself apart from previous studies. Refer to Appendix. C for discussion.

4.2 Implementation of MG-Net

Data encoder in MG-Net. MG-Net exploits three types of data encoder, i.e., the problem encoder, mixer encoder, and depth encoder, which maps the given problem G, the candidate mixer Hamiltonian H_M, and the specified depth p to the same hidden feature space. The construction of these encoders is introduced below and the omitted details are deferred to Appendix D.2.

Cost estimator in MG-Net (Stage 1). Recall Stage 1 in Sec. 4.1, the cost estimator takes the encoded problem graph G_C, the encoded mixer Hamiltonian G_M, and the encoded circuit depth $\boldsymbol{x}_p$ as inputs, and outputs the prediction of the achievable minimum loss of the corresponding QAOA. Each input is processed by an independent branch respectively: *the problem graph branch, the mixer Hamiltonian branch, and the circuit depth branch*, as shown in Fig. 3(a). The concatenation of three types of features is subsequently utilized by a multi-layer perceptron (MLP) to output the minimum loss $\hat{y}$ that the QAOA ansatz can achieve. Refer to Appendix. D.3 for details.

Mixer generator in MG-Net (Stage 2). The mixer generator in MG-Net takes G_C and $\boldsymbol{x}_p$ as input and outputs a targeted mixer Hamiltonian H_M. Specifically, the mixer generation is composed of two separate sub-generators: the operator type generator and the parameter grouping generator defined in Eqn. (5), shown in Fig. 3(b). The operator type generator is responsible for generating operator types $\mathcal{P}$, which is conceptualized as a graph node classification task. The parameter grouping generator is responsible for predicting the sets of index groups $\{\mathcal{G}_j\}_{j=1}^K$ with an unspecified K, which is modeled as a link prediction task. Refer to Appendix. D.3 for details.

4.3 Training strategy

The training process of MG-Net is varied for the first and second stages, under supervised and unsupervised learning paradigms, respectively.

https://doi.org/10.52202/079017-1061

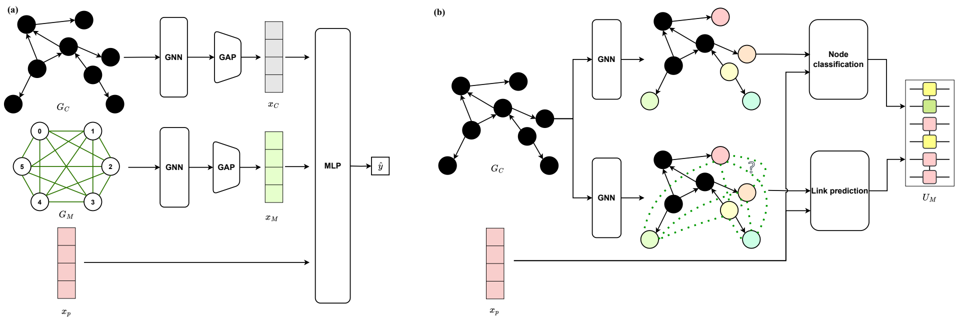

Figure 3: **Structure of cost estimator and mixer generator.** (a) Cost estimator. The cost estimator is comprised of three distinct branches, each dedicated to processing different types of data: the original problem, the candidate mixer Hamiltonian, and the circuit depth. Their outputs are then integrated to predict the cost value achievable by the QAOA circuit. (b) Mixer generator. The mixer generation is divided into two distinct parts: operator type generation and parameter grouping generation. The former is executed as a node classification task, while the latter is approached as a link prediction task.

First-stage training. This stage involves constructing a labeled dataset $\mathcal{D}_{\text{ce}}^{\text{Tr}} = \{(G_C^{(i)}, G_M^{(i)}, \boldsymbol{x}_p^{(i)}), y^{(i)}\}_{i=1}^{S}$, where the i-th sample consists of a tuple of features (i.e., the problem description $G_C^{(i)}$, the mixer $G_M^{(i)}$, and the circuit depth feature $\boldsymbol{x}_p^{(i)}$), and the label $y^{(i)}$ representing the minimum cost value achievable by this QAOA instance (i.e., determined by repeatedly executing such a QAOA with varying initial parameters). Once $\mathcal{D}_{\text{ce}}^{\text{Tr}}$ is ready, the cost estimator is optimized by minimizing the loss function

$$\mathcal{L}_{\text{ce}} = \lambda_e \mathcal{L}_e + \lambda_r \mathcal{L}_r, \tag{6}$$

where $\lambda_e \in [0,1]$ and $\lambda_r \in [0,1]$ are two hyper-parameters of each loss, $\mathcal{L}_e = \frac{1}{S}\sum_{i=1}^{S}(y^{(i)} - \hat{y}^{(i)})^2$ is the mean square error, and $\mathcal{L}_r$ is the ranking loss

$$\mathcal{L}_r = \frac{1}{S^2 - S}\sum_{i,j}^{S} \max(0, 1 - \text{sign}(y^{(i)} - y^{(j)})(\hat{y}^{(i)} - \hat{y}^{(j)})).$$

Second-stage training. This stage involves the training of the mixer generator via unsupervised learning. The loss function of this stage is

$$\mathcal{L}_{\text{mg}} = \frac{1}{S}\sum_{i=1}^{S} C(G_C^{(i)}, M(G_C^{(i)}, \boldsymbol{x}_p^{(i)}), \boldsymbol{x}_p^{(i)}), \tag{7}$$

where $C(\cdot)$ and $M(\cdot)$ represent the output of the cost estimator and mixer generator, respectively. Note that only the parameters of the mixer generator are updated; the cost estimator parameters remain fixed to ensure consistent evaluation criteria throughout the whole learning process.

5 Experiments

We evaluate the performance of MG-Net by two typical applications of QAOA: weighted Max-Cut and Transverse-field Ising model (TFIM), each of which is elucidated below.

Weighted Max-Cut. Denote a weighted graph as $G = (V, E, W)$, where V is the set of vertices of graph, E is the set of graph edges, $W = \{w_{ij}\}_{(i,j)\in E}$ is the set of weights assigned to each edge. The problem Hamiltonian for the weighted Max-Cut problem is $H_C^{\text{MaxCut}} = 0.5 * \sum_{(i,j)\in E} w_{ij} Z_i Z_j$, where Z_i is a Pauli-Z operator acting on the i-th qubit.

TFIM. Our focus is a class of inhomogeneous TFIMs: $H_C^{\text{TFIM}} = -\sum_{(i,j)} J_{ij} Z_i Z_j - h \sum_i X_i$, where J_{ij} is the interaction strength between neighboring spins (or qubits) (i,j), and h signifies the strength of a global transverse field applied to each spin. In this model, the interaction strengths J_{ij} can vary between different pairs of spins, adding a layer of complexity to the system.

5.1 Experiment configuration

Dataset construction. The Max-Cut problem focuses weighted 3-degree regular (w3r) graphs, where the edge weights $\{w_{ij}\}$ are uniformly sampled from $[0, 1]$. The TFIM focuses on 1D instances where a qubit $i \in [N-1]$ has neighbors $i \pm 1 \pmod N$. The strength J_{ij} and h are uniformly sampled from $[0.5, 1.5]$ and $[0.1, 2]$ respectively. The training dataset $\mathcal{D}_{\mathrm{ce}}^{\mathrm{Tr}}$ in Sec. 4.3 contains $S = 100$ instances for both two tasks with size up to $N = 64$ qubits, while The test dataset $\mathcal{D}^{\mathrm{Te}}$ contains another 100 problem instances which are different from that of $\mathcal{D}_{\mathrm{ce}}^{\mathrm{Tr}}$. Refer to Appendix D.1 for details.

Optimization and training of MG-Net. The cost estimator and mixer generator are trained using an Adam optimizer with a learning rate of 10^{-4}, and hyper-parameters $\lambda_e = 1$ and $\lambda_r = 1$ in Eqn. (6).

Optimization of QAOA. After predicting the problem-hardware-tailored mixer Hamiltonian H_M by the trained mixer generator, a QAOA circuit with the initial state $|+\rangle^{\otimes N}$ and H_M is optimized by an Adam optimizer with a learning rate of 0.15. Each setting undergoes 10 independent runs with varied random seeds and initial parameters to obtain the statistical results. Refer to Appendix D.4 for detailed discussion about the selection of initial state.

5.2 Results

Cost estimator acts as an accurate performance indication for QAOA. The behavior of the cost estimator on the test dataset with varying circuit depths p and two distinct parameter grouping strategies NG and FG (defined in Theorem 3.1) is recorded in Fig. 4. In Fig. 4(a), we observed a strong correlation between the estimated and minimum cost values, and the correlation strength changes with p and parameter grouping strategy. Particularly, the cost estimator predicts a high likelihood of finding the most accurate solution for QAOA circuits with FG parameters and a depth of $p = 92$. This prediction aligns with the actual performance of QAOA under these specific conditions. To further investigate the cost estimator's behavior with more complex mixer operator types, we conducted additional experiments using an extended set of candidate operators $\{X, Y, XX, YY\}$, which includes two-qubit operators. As shown in Fig. 4 (b), the cost estimator continues to demonstrate its efficiency and high performance, even as the complexity of the mixer Hamiltonian design increases.

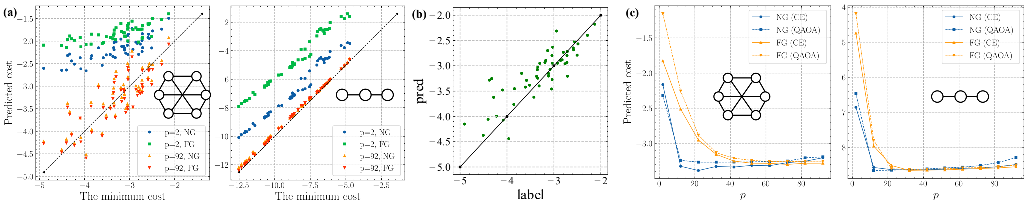

Figure 4: **Behavior of cost estimator**. (a) The correlation between the estimated cost and the minimum cost for Max-Cut (left) and TFIM (right). Each point represents the result of a problem instance. The dashed line represents that QAOA can find the exact solution $y = x$. (b) Behavior of cost estimator with extended mixer operator pool $\{X, Y, XX, YY\}$. 'label' represents the actual achieved approximation ratio, while 'pred' represents the result predicted by the cost estimator. (c) The achievable cost under various circuit depth p for Max-Cut (left) and TFIM (right). The label 'CE' is the abbreviation of cost estimator. The dashed lines represent the cost achieved by QAOA, while the solid lines represent the cost estimated by our model.

We next focus on the behavior of the cost estimator concerning p as shown in Fig. 4(c). We note that for FG (standard QAOA), the estimated loss decreased monotonically with increasing p, aligning with standard QAOA's behavior. Under the NG scenario (multi-angle QAOA), a transition that QAOA performance begins to decline is observed when the circuit becomes excessively long ($p > 42$). These results indicate the reliability of the cost estimator as a performance indicator for QAOA and reveal the complexities in QAOA performance under conditions of increased circuit length.

Mixer generator. We next evaluate the performance of the customized mixer Hamiltonian generated by MG-Net. As shown in Fig. 5(a), the number of trainable parameters $\#P$ of the generated quantum circuits aligns with the maximum in scenarios where all parameters are non-correlated (labeled as 'NG') for smaller circuit depths $p < 20$. This alignment indicates that MG-Net effectively enhances

https://doi.org/10.52202/079017-1061

the expressibility of the QAOA ansatz for limited-depth circuits without significantly increasing the number of parameters, thereby avoiding potential trainability issues. As p increases, a transition occurs. The growth rate of $\#P$ starts to decelerate, reaching a notable transition point at $p = 62$ for Max-Cut ($p = 52$ for TFIM). Beyond this threshold, the generated mixer Hamiltonians gradually converge towards the configuration seen in standard QAOA, with fully grouped parameters.

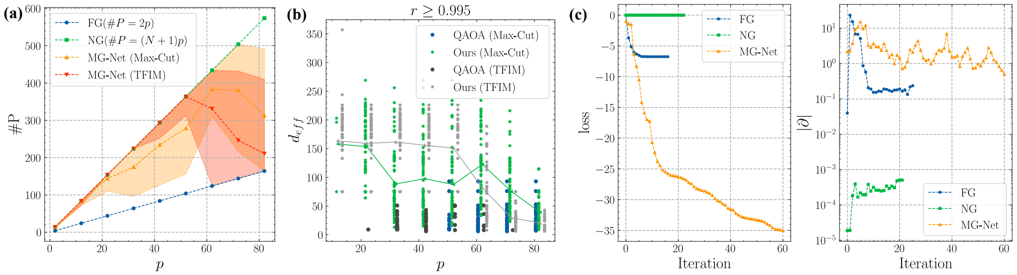

Figure 5: **The trainability of the quantum circuits generated by MG-Net for Max-Cut and TFIM.** (a) The number $\#P$ of trainable parameters of the quantum circuits with mixer Hamiltonian predicted by MG-Net. (b) Comparison of the effective dimension d_{eff} of quantum circuits in standard QAOA and MG-Net driven QAOA (labeled as 'Ours'). The green and grey solid lines denote the average effective dimension d_{eff} of the predicted circuits that can achieve an approximation ratio over 0.995 for Max-Cut and TFIM, respectively. It assesses circuits achieving an approximation ratio r of at least 0.995. (c) The convergence of QAOA with FG, NG and mixer Hamiltonian predicted by MG-Net for Max-Cut on 64-node weighted graphs.

Fig. 5(b) compares the effective dimension d_{eff} of quantum circuits achieving high approximation ratio $r \geq 0.995$ in standard QAOA and MG-Net driven QAOA. The results show that circuits generated by MG-Net achieve $r \geq 0.995$ across all values of p, even as low as $p = 2$, outperforming standard QAOA, which only reaches this level for $p > 50$ for Max-Cut ($p > 20$ for TFIM). Besides, the effective dimension of these high-quality quantum circuits gradually decreases with growing p, in line with the convergence analysis in Theorem 3.1. These findings suggest that MG-Net dynamically adjusts quantum circuits in response to changes in circuit depth p, thereby consistently ensuring high performance.

Fig. 5(c) explicitly demonstrates the optimization behavior of 64-qubit QAOA with FG, NG and the mixer Hamiltonian predicted by our MG-Net. The left panel displays the loss curves during the optimization of quantum circuits with $p = 2$, revealing that our method achieves the most rapid convergence. The right panel further explores the gradients of the three methods during optimization. Notably, the parameter gradient norm of our method maintains a trainable level of 1, whereas the gradient for FG and NG falls to 10^{-1} and 10^{-4}, respectively, compromising their trainability.

Performance comparison. In evaluating the effectiveness of our proposed method for solving Max-Cut problems, we conducted a comparative analysis against both classical and quantum algorithms. The benchmarks included the greedy algorithm, the Goemans-Williamson (GW) algorithm [35], alongside various quantum approaches such as QAOA, ADAPT-QAOA, and multi-angle QAOA (ma-QAOA). Our analysis, based on the average results from 100 graphs in our test dataset, is summarized in Tab. 2. The findings reveal that our method consistently outperforms other techniques in achieving a higher approximation ratio, particularly in larger-scale problems. Refer to Appendix E.1 for comparison results on TFIM.

More numerical results. We have conducted additional analysis on the behavior of MG-Net and additional experiments on more tasks. Refer to E for more details.

6 Conclusion

In this study, we analyze QAOA's convergence on varied mixer Hamiltonians, focusing on parameter grouping strategies. We introduce MG-Net for dynamically generating optimal mixer Hamiltonians for various problems and circuit depths. Numerical experiments on Max-Cut and TFIM confirm MG-Net's efficacy in enhancing QAOA's approximation ratio, particularly for large-scale problems,

https://doi.org/10.52202/079017-1061

Table 2: **Comparison of approximation ratio r among different methods for Max-Cut.**

Method	6 qubits	16 qubits	64 qubits
Greedy	0.89 ± 0.104	0.91 ± 0.047	0.79
GW	0.94 ± 0.074	0.93 ± 0.052	0.91
QAOA	0.93 ± 0.027	0.35 ± 0.119	0.19
ADAPT-QAOA	0.75 ± 0.129	0.58 ± 0.154	-
ma-QAOA	0.98 ± 0.004	0.84 ± 0.129	0.0
Ours	$\mathbf{0.99 \pm 0.0004}$	$\mathbf{0.95 \pm 0.152}$	**0.96**

while ensuring low circuit complexity. This research advances the understanding and application of QAOA across various circuit depths.

Despite these promising outcomes, our work has several limitations that need to be addressed in future research. Firstly, training the cost estimator of MG-Net involves the construction of a labeled dataset $\mathcal{D}_{\mathrm{ce}}^{\mathrm{Tr}}$, which introduces additional resource consumption. Future work can focus on more efficient training algorithms. Additionally, our current approach is specifically designed for QAOA on early fault-tolerant devices, which limits the exploration of extending MG-Net to other quantum algorithms and noisy devices. Addressing these limitations will further enhance the robustness and scalability of MG-Net, offering potential for broader use in VQAs.

Acknowledgments and Disclosure of Funding

The proposed research is partially supported by Dr Tao's NTU RSR and Start Up Grants.

References

[1] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012.

[2] Clayton W Commander. Maximum cut problem, max-cut. *Encyclopedia of Optimization*, 2, 2009.

[3] Tommy R Jensen and Bjarne Toft. *Graph coloring problems*. John Wiley & Sons, 2011.

[4] Karla L Hoffman, Manfred Padberg, Giovanni Rinaldi, et al. Traveling salesman problem. *Encyclopedia of operations research and management science*, 1:1573–1578, 2013.

[5] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[6] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010.

[7] Andrew Lucas. Ising formulations of many np problems. *Frontiers in physics*, 2:5, 2014.

[8] Young-Hyun Oh, Hamed Mohammadbagherpoor, Patrick Dreher, Anand Singh, Xianqing Yu, and Andy J Rindos. Solving multi-coloring combinatorial optimization problems using hybrid quantum algorithms. *arXiv preprint arXiv:1911.00595*, 2019.

[9] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[10] E Farhi and AW Harrow. Quantum supremacy through the quantum approximate optimization algorithm (2016). *arXiv preprint arXiv:1602.07674*.

[11] Seth Lloyd. Quantum approximate optimization is computationally universal. *arXiv preprint arXiv:1812.11075*, 2018.

[12] Mauro ES Morales, Jacob D Biamonte, and Zoltán Zimborás. On the universality of the quantum approximate optimization algorithm. *Quantum Information Processing*, 19:1–26, 2020.

[13] Kostas Blekos, Dean Brand, Andrea Ceschini, Chiao-Hui Chou, Rui-Hao Li, Komal Pandya, and Alessandro Summer. A review on quantum approximate optimization algorithm and its variants. *Physics Reports*, 1068:1–66, 2024.

[14] Zhihui Wang, Stuart Hadfield, Zhang Jiang, and Eleanor G Rieffel. Quantum approximate optimization algorithm for maxcut: A fermionic view. *Physical Review A*, 97(2):022304, 2018.

[15] Guido Pagano, Aniruddha Bapat, Patrick Becker, Katherine S Collins, Arinjoy De, Paul W Hess, Harvey B Kaplan, Antonis Kyprianidis, Wen Lin Tan, Christopher Baldwin, et al. Quantum approximate optimization of the long-range ising model with a trapped-ion quantum simulator. *Proceedings of the National Academy of Sciences*, 117(41):25396–25401, 2020.

[16] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Physical Review X*, 10(2):021067, 2020.

[17] Rebekah Herrman, Phillip C Lotshaw, James Ostrowski, Travis S Humble, and George Siopsis. Multi-angle quantum approximate optimization algorithm. *Scientific Reports*, 12(1):6781, 2022.

[18] Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, Daniel J Egger, Stefan Filipp, Andreas Fuhrer, Jay M Gambetta, Marc Ganzhorn, et al. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology*, 3(3):030503, 2018.

[19] Gian Giacomo Guerreschi and Anne Y Matsuura. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific reports*, 9(1):6903, 2019.

[20] Michael Victor Berry. Transitionless quantum driving. *Journal of Physics A: Mathematical and Theoretical*, 42(36):365303, 2009.

[21] David Guéry-Odelin, Andreas Ruschhaupt, Anthony Kiely, Erik Torrontegui, Sofia Martínez-Garaot, and Juan Gonzalo Muga. Shortcuts to adiabaticity: Concepts, methods, and applications. *Reviews of Modern Physics*, 91(4):045001, 2019.

[22] Yunlong Yu, Chenfeng Cao, Carter Dewey, Xiang-Bin Wang, Nic Shannon, and Robert Joynt. Quantum approximate optimization algorithm with adaptive bias fields. *Physical Review Research*, 4(2):023249, 2022.

[23] Frederic Sauvage, Martin Larocca, Patrick J Coles, and Marco Cerezo. Building spatial symmetries into parameterized quantum circuits for faster training. *Quantum Science and Technology*, 2022.

[24] Edwin Williams. *Representation theory*. MIT Press, 2002.

[25] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[26] Yang Qian, Xinbiao Wang, Yuxuan Du, Xingyao Wu, and Dacheng Tao. The dilemma of quantum neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[27] Xuchen You, Shouvanik Chakrabarti, and Xiaodi Wu. A convergence theory for over-parameterized variational quantum eigensolvers. *arXiv preprint arXiv:2205.12481*, 2022.

[28] Xinbiao Wang, Junyu Liu, Tongliang Liu, Yong Luo, Yuxuan Du, and Dacheng Tao. Symmetric pruning in quantum neural networks, 2023.

[29] Martin Larocca, Piotr Czarnik, Kunal Sharma, Gopikrishnan Muraleedharan, Patrick J Coles, and M Cerezo. Diagnosing barren plateaus with tools from quantum optimal control. *Quantum*, 6:824, 2022.

[30] Martin Larocca, Nathan Ju, Diego García-Martín, Patrick J Coles, and Marco Cerezo. Theory of overparametrization in quantum neural networks. *Nature Computational Science*, 3(6):542–551, 2023.

[31] Xuchen You and Xiaodi Wu. Exponentially many local minima in quantum neural networks. In *International Conference on Machine Learning*, pages 12144–12155. PMLR, 2021.

[32] Eric Ricardo Anschuetz. Critical points in quantum generative models, 2022.

[33] Pranav Chandarana, Narendra N Hegade, Koushik Paul, Francisco Albarrán-Arriagada, Enrique Solano, Adolfo Del Campo, and Xi Chen. Digitized-counterdiabatic quantum approximate optimization algorithm. *Physical Review Research*, 4(1):013141, 2022.

[34] Linghua Zhu, Ho Lun Tang, George S Barron, FA Calderon-Vargas, Nicholas J Mayhall, Edwin Barnes, and Sophia E Economou. Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer. *Physical Review Research*, 4(3):033029, 2022.

[35] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[36] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. Quantum circuit learning. *Physical Review A*, 98(3):032309, 2018.

[37] Louis Schatzki, Martin Larocca, Quynh T Nguyen, Frederic Sauvage, and Marco Cerezo. Theoretical guarantees for permutation-equivariant quantum neural networks. *npj Quantum Information*, 10(1):12, 2024.

[38] Barry Simon. *Representations of finite and compact groups*. Number 10. American Mathematical Soc., 1996.

[39] Ruslan Shaydulin and Stefan M Wild. Exploiting symmetry reduces the cost of training qaoa. *IEEE Transactions on Quantum Engineering*, 2:1–9, 2021.

[40] Kaiyan Shi, Rebekah Herrman, Ruslan Shaydulin, Shouvanik Chakrabarti, Marco Pistoia, and Jeffrey Larson. Multiangle qaoa does not always need all its angles. In *2022 IEEE/ACM 7th Symposium on Edge Computing (SEC)*, pages 414–419. IEEE, 2022.

[41] Michelle Chalupnik, Hans Melo, Yuri Alexeev, and Alexey Galda. Augmenting qaoa ansatz with multiparameter problem-independent layer. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 97–103. IEEE, 2022.

[42] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.

[43] Takuya Yoshioka, Keita Sasada, Yuichiro Nakano, and Keisuke Fujii. Fermionic quantum approximate optimization algorithm. *Physical Review Research*, 5(2):023071, 2023.

[44] Jonathan Wurtz and Peter J Love. Counterdiabaticity and the quantum approximate optimization algorithm. *Quantum*, 6:635, 2022.

[45] Andreas Bärtschi and Stephan Eidenbenz. Grover mixers for qaoa: Shifting complexity from mixer design to state preparation. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 72–82. IEEE, 2020.

[46] Sergey Bravyi, Alexander Kliesch, Robert Koenig, and Eugene Tang. Obstacles to variational quantum optimization from symmetry protection. *Physical review letters*, 125(26):260505, 2020.

[47] Javier Villalba-Diez, Ana González-Marcos, and Joaquín B Ordieres-Meré. Improvement of quantum approximate optimization algorithm for max–cut problems. *Sensors*, 22(1):244, 2021.

[48] Shi-Xin Zhang, Chang-Yu Hsieh, Shengyu Zhang, and Hong Yao. Neural predictor based quantum architecture search. *Machine Learning: Science and Technology*, 2(4):045027, 2021.

[49] Esther Ye and Samuel Yen-Chi Chen. Quantum architecture search via continual reinforcement learning. *arXiv preprint arXiv:2112.05779*, 2021.

[50] Mateusz Ostaszewski, Lea M Trenkwalder, Wojciech Masarczyk, Eleanor Scerri, and Vedran Dunjko. Reinforcement learning for optimization of variational quantum circuit architectures. *Advances in Neural Information Processing Systems*, 34:18182–18194, 2021.

[51] En-Jui Kuo, Yao-Lung L Fang, and Samuel Yen-Chi Chen. Quantum architecture search via deep reinforcement learning. *arXiv preprint arXiv:2104.07715*, 2021.

[52] Fan-Xu Meng, Ze-Tong Li, Xu-Tao Yu, and Zai-Chen Zhang. Quantum circuit architecture optimization for variational quantum eigensolver via monto carlo tree search. *IEEE Transactions on Quantum Engineering*, 2:1–10, 2021.

[53] Yuxuan Du, Tao Huang, Shan You, Min-Hsiu Hsieh, and Dacheng Tao. Quantum circuit architecture search for variational quantum algorithms. *npj Quantum Information*, 8(1):62, 2022.

[54] Kehuan Linghu, Yang Qian, Ruixia Wang, Meng-Jun Hu, Zhiyuan Li, Xuegang Li, Huikai Xu, Jingning Zhang, Teng Ma, Peng Zhao, et al. Quantum circuit architecture search on a superconducting processor. *arXiv preprint arXiv:2201.00934*, 2022.

[55] Zhimin He, Chuangtao Chen, Lvzhou Li, Shenggen Zheng, and Haozhen Situ. Quantum architecture search with meta-learning. *Advanced Quantum Technologies*, 5(8):2100134, 2022.

[56] Shi-Xin Zhang, Chang-Yu Hsieh, Shengyu Zhang, and Hong Yao. Differentiable quantum architecture search. *Quantum Science and Technology*, 7(4):045023, 2022.

[57] Wenjie Wu, Ge Yan, Xudong Lu, Kaisen Pan, and Junchi Yan. Quantumdarts: differentiable quantum architecture search for variational quantum algorithms. In *International Conference on Machine Learning*, pages 37745–37764. PMLR, 2023.

[58] Cong Lei, Yuxuan Du, Peng Mi, Jun Yu, and Tongliang Liu. Neural auto-designer for enhanced quantum kernels. In *The Twelfth International Conference on Learning Representations*, 2024.

[59] Xudong Lu, Kaisen Pan, Ge Yan, Jiaming Shan, Wenjie Wu, and Junchi Yan. Qas-bench: rethinking quantum architecture search and a benchmark. In *International Conference on Machine Learning*, pages 22880–22898. PMLR, 2023.

[60] Linghua Zhu, Ho Lun Tang, George S Barron, FA Calderon-Vargas, Nicholas J Mayhall, Edwin Barnes, and Sophia E Economou. An adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer. *arXiv preprint arXiv:2005.10258*, 2020.

[61] Zeqiao Zhou, Yuxuan Du, Xinmei Tian, and Dacheng Tao. Qaoa-in-qaoa: solving large-scale maxcut problems on small quantum machines. *Physical Review Applied*, 19(2):024027, 2023.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[63] Yang Qian, Yuxuan Du, Zhenliang He, Min-Hsiu Hsieh, and Dacheng Tao. Multimodal deep representation learning for quantum cross-platform verification. *Physical Review Letters*, 133(13):130601, 2024.

[64] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Shahnawaz Ahmed, Vishnu Ajith, M Sohaib Alam, Guillermo Alonso-Linaje, B AkashNarayanan, Ali Asadi, et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*, 2018.

https://doi.org/10.52202/079017-1061

[65] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

A Optimization of QAOA

In this section, we separately elaborate on the elementary notations in quantum computing, the preliminary of Hamiltonian, and the optimization strategy of QAOA.

Basics of quantum computation. The elementary unit of quantum computation is qubit (or quantum bit), which is the quantum mechanical analog of a classical bit. A qubit is a two-level quantum-mechanical system described by a unit vector in the Hilbert space $\mathbb{C}^2$. In Dirac notation, a qubit state is defined as $|\phi\rangle = c_0 |0\rangle + c_1 |1\rangle \in \mathbb{C}^2$ where $|0\rangle = [1,0]^\top$ and $|1\rangle = [0,1]^T$ specify two unit bases and the coefficients $c_0, c_1 \in \mathbb{C}$ yield $|c_0|^2 + |c_1|^2 = 1$. Similarly, the *quantum state* of n qubits is defined as a unit vector in $\mathbb{C}^{2^n}$, i.e., $|\psi\rangle = \sum_{j=1}^{2^n} c_j |e_j\rangle$, where $|e_j\rangle \in \mathbb{R}^{2^n}$ is the computational basis whose j-th entry is 1 and other entries are 0, and $\sum_{j=1}^{2^n} |c_j|^2 = 1$ with $c_j \in \mathbb{C}$. Besides Dirac notation, the density matrix can be used to describe more general qubit states. For example, the density matrix of the state $|\psi\rangle$ is $\rho = |\psi\rangle \langle\psi| \in \mathbb{C}^{2^n \times 2^n}$, where $\langle\psi| = |\psi\rangle^\dagger$ refers to the complex conjugate transpose of $|\psi\rangle$. For a set of qubit states $\{p_j, |\psi_j\rangle\}_{j=1}^m$ with $p_j > 0$, $\sum_{j=1}^m p_j = 1$, and $|\psi_j\rangle \in \mathbb{C}^{2^n}$ for $j \in [m]$, its density matrix is $\rho = \sum_{j=1}^m p_j \rho_j$ with $\rho_j = |\psi_j\rangle \langle\psi_j|$ and $\mathrm{Tr}(\rho) = 1$.

A *quantum gate* is a unitary operator that can evolve a quantum state ρ to another quantum state ρ'. Namely, an n-qubit gate $U \in \mathcal{U}(2^n)$ obeys $UU^\dagger = U^\dagger U = I_{2^n}$, where $\mathcal{U}(2^n)$ refers to the unitary group in dimension 2^n. Typical single-qubit quantum gates include the Pauli gates, which can be written as Pauli matrices:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{8}$$

The more general quantum gates are their corresponding rotation gates $R_X(\theta) = e^{-i\frac{\theta}{2}X}$, $R_Y(\theta) = e^{-i\frac{\theta}{2}Y}$, and $R_Z(\theta) = e^{-i\frac{\theta}{2}Z}$ with a tunable parameter θ, which can be written in the matrix form as

$$R_X(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}, R_Y(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}, R_Z(\theta) = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}. \tag{9}$$

They are equivalent to rotating a tunable angle θ around x, y, and z axes of the Bloch sphere, and recovering the Pauli gates X, Y, and Z when $\theta = \pi$. Moreover, a multi-qubit gate can be either an individual gate (e.g., CNOT gate) or a tensor product of multiple single-qubit gates.

The *quantum measurement* refers to the procedure of extracting classical information from the quantum state. It is mathematically specified by a Hermitian matrix H called the *observable*. Applying the observable H to the quantum state $|\psi\rangle$ yields a random variable whose expectation value is $\langle\psi| H |\psi\rangle$.

Hamiltonian and ground state. In quantum computation, a *Hamiltonian* is a Hermitian matrix that is used to characterize the evolution of a quantum system or as an observable to extract the classical information from the quantum system. Specifically, under the Schrödinger equation, a quantum gate has the mathematical form of $U = e^{-itH}$, where H is a Hermitian matrix, called the Hamiltonian of the quantum system, and t refers to the evolution time of the Hamiltonian. Typical single-qubit Hamiltonians include the Pauli matrices defined in Eqn. (8). As a result, the evolution time t refers to the tunable parameter θ in Eqn. (9). Any single-qubit Hamiltonian can be decomposed as the linear combination of Pauli matrices, i.e., $H = a_1 I + a_2 X + a_3 Y + a_4 Z$ with $a_j \in \mathbb{C}$. In the same way, a multi-qubit Hamiltonian is denoted by $H = \sum_{j=1}^{4^n} a_j P_j$, where $P_j \in \{I, X, Y, Z\}^{\otimes n}$ is the tensor product of Pauli matrices. In quantum chemistry and quantum many-body physics, the Hermitian matrix that describes the quantum system to be solved is denoted as the *problem Hamiltonian H_C*. Within the context of QAOA, the information of the graph is encoded in the problem Hamiltonian, which is also called cost Hamiltonian. Another essential Hamiltonian in QAOA refers to the mixer

Hamiltonian H_M, which is designed to facilitate transitions between different states (solutions), allowing the algorithm to explore the solution space.

When taking the problem Hamiltonian as the observable, the quantum state $|\psi^*\rangle$ is said to be the *ground state* of problem Hamiltonian H if the expectation value $\langle\psi^*| H |\psi^*\rangle$ takes the minimum eigenvalue of H, which is called the *ground energy*. The solution of the optimization problem is encoded in the ground state of the problem Hamiltonian.

Optimization of QAOA. The loss function for QAOA with problem Hamiltonian H_C is generally defined as

$$\mathcal{L}(\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta})) = \langle\psi_0|U(\boldsymbol{\theta})^\dagger H_C U(\boldsymbol{\theta})|\psi_0\rangle\,, \tag{10}$$

where $U(\boldsymbol{\theta})$ refers to the parameterized unitary implemented on a quantum computer and $|\psi_0\rangle$ is an easily prepared state, which is generally set as the computational basis state $|0^{\otimes n}\rangle$. The optimization of the loss function $\mathcal{L}(\boldsymbol{\theta})$ can be completed by gradient-based methods. A plethora of optimizers have been designed to estimate the optimal parameters $\boldsymbol{\theta}^* = \min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$. Here we introduce the implementation of the first-order gradient-based optimizer for self-consistency. Refer to [25] for a comprehensive review.

Based on Eqn. (1), the trainable parameters of QAOA are denoted by $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_L^\top)^\top$ with $\boldsymbol{\theta}_\ell = (\theta_{\ell 1}, \cdots, \theta_{\ell K})^T$, where the subscript '$\ell k$' refers to the k-th parameter of the ℓ-th layer U_ℓ for $\forall k \in [K]$ and $\forall \ell \in [L]$. The corresponding update rule at the t-th iteration $\forall t \in [T]$ is

$$\begin{aligned}
& \boldsymbol{\theta}^{(t+1)} \\
= \quad & \boldsymbol{\theta}^{(t)} - \eta \frac{\partial \mathcal{L}(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}} \\
= \quad & \boldsymbol{\theta}^{(t)} - \eta \left(\langle\psi_0| U(\boldsymbol{\theta}^{(t)})^\dagger H_C U(\boldsymbol{\theta}^{(t)}) |\psi_0\rangle - E_0 \right) \frac{\partial \left(\langle\psi_0| U(\boldsymbol{\theta}^{(t)})^\dagger H_C U(\boldsymbol{\theta}^{(t)}) |\psi_0\rangle - E_0 \right)}{\partial \boldsymbol{\theta}},
\end{aligned}$$

where η refers to the learning rate. The derivative in the last equality can be calculated via the parameter shift rule [36]. Mathematically, the derivative with respect to the parameter $\theta_{\ell k}$ for $\forall \ell \in [L]$ and $\forall k \in [K]$ is

$$\begin{aligned}
& \frac{\partial \left(\langle\psi_0| U(\boldsymbol{\theta})^\dagger H_C U(\boldsymbol{\theta}) |\psi_0\rangle - E_0 \right)}{\partial \theta_{\ell k}} \\
= \quad & \frac{1}{2\sin\alpha} \Big[\left(\langle\psi_0| U(\boldsymbol{\theta}^+)^\dagger H_C U((\boldsymbol{\theta}^+) |\psi_0\rangle - E_0 \right) - \left(\langle\psi_0| U((\boldsymbol{\theta}^-)^\dagger H_C U((\boldsymbol{\theta}^-) |\psi_0\rangle - E_0 \right) \Big],
\end{aligned}$$

where $\boldsymbol{\theta}^+ = \boldsymbol{\theta} + \alpha \boldsymbol{e}_{\ell k}$, $\boldsymbol{\theta}^- = \boldsymbol{\theta} - \alpha \boldsymbol{e}_{\ell k}$, $\boldsymbol{e}_{\ell k}$ is the unit vector along the $\theta_{\ell k}$ axis and α can be any real number but the multiple of π because of the diverging denominator.

B Proof

The theoretical analysis of the convergence for symmetric QAOA is based on representation theory. In this regard, we first introduce the foundation of representation theory related to QAOA in Appendix B.1. The proof of Theorem 3.1 is elaborated in Appendix B.2.

B.1 Representation theory in QAOA

In general, an instance of QAOA is specified by a triplet $(|\psi_0\rangle, U(\boldsymbol{\theta}), H)$, where $|\psi_0\rangle$ and H refer to the initial state and problem Hamlitonian, and $U(\boldsymbol{\theta})$ refers to the parameterized quantum circuit (ansatz) with the form of

$$U(\boldsymbol{\theta}) = \prod_{j=1}^{P} \prod_{k=1}^{K} e^{-i\theta_{j,k} H_k}, \tag{11}$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_{11}, \cdots, \boldsymbol{\theta}_{1K}, \cdots, \boldsymbol{\theta}_{P1}, \cdots, \boldsymbol{\theta}_{PK}) \in \Theta \subseteq \mathbb{R}^{PK}$ is trainable parameters, j is the index of layer, and $\mathcal{A} = \{H_k\}_{k=1}^K$ is set of Hermitian traceless operators called an ansatz design. The difference of ansatz originates from the varied Θ and $\mathcal{A}$. Given Θ and $\mathcal{A}$, a set of ansatz forms a subgroup of $SU(2^n)$ with $\mathcal{U}_{\mathcal{A}} = \cup_{L=0}^{\infty}\{U(\boldsymbol{\theta}) : \boldsymbol{\theta} \in \Theta\}$, which can be characterized by dynamical Lie group with dynamical Lie algebra [29]

Definition B.1 (Dynamical Lie algebra and dynamical Lie group, [29])**.** Given an ansatz design $\mathcal{A} = \{H_1, \cdots, H_K\}$, the dynamical Lie algebra (DLA) $\mathfrak{g}$ is generated by the repeated nested commutators of elements in $\mathcal{A}$, i.e.,

$$\mathfrak{g} = \mathrm{span}\, \langle iH_1, ..., iH_K \rangle_{Lie}\,, \tag{12}$$

where $\langle S \rangle_{Lie}$ denotes the Lie closure, i.e., the set obtained by repeatedly taking the nested commutators of the elements in S. The set of unitaries $\mathcal{U}_{\mathcal{A}}$ that can be generated by the ansatz design $\mathcal{A}$ is determined by its DLA through

$$\mathcal{U}_{\mathcal{A}} = e^{\mathfrak{g}} := \{e^H, H \in \mathfrak{g}\}. \tag{13}$$

Furthermore, the algebra structures of the ansatz design $\mathcal{A}$ can be characterized through the representation and the subrepresentation of Lie algebra on specific vector space.

Definition B.2 (Representation of Lie algebra)**.** Let $\mathfrak{g}$ be a Lie algebra on a finite-dimensional vector space V. A representation r of $\mathfrak{g}$ acting on V is a Lie algebra homomorphism $r : \mathfrak{g} \to \mathfrak{gl}(V)$, i.e., a linear map satisfying

$$r([X, Y]) = [r(X), r(Y)], \quad \text{for all } X, Y \in \mathfrak{g}. \tag{14}$$

The dimension of the representation r is defined by $\dim(r) = \dim(V)$. If there exists a direct sum decomposition of V into subspaces $V = V_1 \oplus V_2 \oplus \cdots \oplus V_k$ such that $r(g)v_j \in V_j$ for any $v_j \in V_j$ and any $g \in \mathfrak{g}$, then $r_j := r|_{V_j}$ is called the subrepresentation of r on the vector space V_j. Moreover, r_j is irreducible if there is no non-trivial invariant subspace of V_j. Then the representation of $\mathfrak{g}$ on the vector space $V = V_1 \oplus V_2 \oplus \cdots \oplus V_k$ can be written as

$$r(g)(v) = (r_1 \oplus \cdots \oplus r_k(g))(v_1, \cdots, v_k) = (r_1(g)v_1, \cdots, r_k(g)v_k), \quad \text{for all } g \in \mathfrak{g},\ v \in V. \tag{15}$$

The dimension of the representation with irreducible representation in Eqn. (15) is $\dim(r) = \sum_{j=1}^{k} \dim(V_j)$

The representation of DLA $\mathfrak{g}$ refers to the natural representation $r : \mathfrak{g} \to \mathfrak{g}$. In this regard, the dimension of DLA refers to $\dim(\mathfrak{g}) = \dim(r)$. While the dimension of DLA is employed to characterize the threshold of over-parameterization [30] and the barren plateau [29], it does not take into account the symmetry structure of the ansatz and the initial state concerning the problem Hamiltonian. In particular, the symmetry operators of the DLA $\mathfrak{g}$ refer to unitary operators S satisfying $SgS^\dagger = g$ for any $g \in \mathfrak{g}$, which is a subset of the commutant of $\mathfrak{g}$.

Definition B.3 (Commutant)**.** Let $\mathfrak{g}$ be a matrix algebra. Its commutant is defined as $\mathcal{C}(\mathfrak{g}) := \{A : [A, g] = 0, \forall g \in \mathfrak{g}\}$.

We recall that the ansatz being symmetric with respect to the problem Hamiltonian means that there exists a symmetry group of the problem Hamiltonian $\mathcal{S} = \{S : S^\dagger H_C S = H_C\}$ such that $\mathcal{S}$ is also the symmetry group of the related DLA $\mathfrak{g}$, i.e., $\mathcal{S} \subseteq \mathcal{C}(\mathfrak{g})$. This indicates that the problem Hamiltonian and the ansatz design have the same block diagonalization structure [37], namely the acting vector space $V = \oplus_{j=1}^{k} V_j$. Moreover, when there exists a subspace $V^* \in \{V_j\}_{j=1}^{k}$ such that the initial state lives in this space, then the optimization of the variational quantum state could be constrained into this subspace V^* whose dimension refers to the effective dimension defined in Definition 2.1. In this regard, the trainability of QAOA could be instead characterized by the effective dimension $d_{\text{eff}} = \dim(V^*)$ [28, 27]. The relation between the effective dimension and the dimension of DLA is encapsulated in the following lemma.

Lemma B.4 (The relation between effective dimension and the dimension of DLA)**.** *Consider a QAOA instance ($|\psi_0\rangle$, $U(\boldsymbol{\theta}), H_P$) with DLA $\mathfrak{g}$. If there exists an invariant subspace $V_{\mathfrak{g}}$ covering the initial state $|\psi_0\rangle$ and the solution state $|\psi^*\rangle = U(\boldsymbol{\theta}^*) |\psi_0\rangle$, then the effective dimension d_{eff} of this ansatz design $\mathcal{A}$ and the dimension of the corresponding DLA $\mathfrak{g}$ yields $d_{\text{eff}} \leq \dim(\mathfrak{g})$.*

Proof of Lemma B.4. The derivation of $d_{\text{eff}} \leq \dim(\mathfrak{g})$ could be directly obtained from the observation of $d_{\text{eff}} \leq \max_{j \in [k]} \dim(V_j) \leq \sum_{j=1}^{k} V_j = \dim(\mathfrak{g})$. □

B.2 Proof of Theorem 3.1

The proof of Theorem 3.1 employs the following lemmas, whose proofs are deferred to Appendix B.3.

Lemma B.5 (Convergence, adapted from Corollary 5.4 in [27]). *Consider a QAOA instance denoted as ($|\psi_0\rangle, U(\boldsymbol{\theta}), H_C$) with the effective dimension d_{eff}. The unitary operator $U(\boldsymbol{\theta})$ follows the Haar distribution over special unitary matrices. Let $|\psi^*\rangle$ denote the solution state for problem Hamiltonin H_C and $|\psi^{(t)}\rangle$ be the state at the t-th iteration. There exists an d_{eff}-dependent over-parameter threshold $C(d_{\text{eff}})$ and a PK-dependent learning rate $\eta(PK)$ so that if the number of the ansatz parameters $PK \geq C$, then with high probability, under gradient flow with learning rate $\eta(PK)$, the output state $|\psi^{(t)}\rangle$ converges to the solution state with error $\epsilon = 1 - |\langle\psi^{(t)}|\psi^*\rangle|$ after $T_\epsilon = O(\log \frac{d_{\text{eff}}}{\epsilon})$ iterations.*

Lemma B.6. *Let $\mathcal{A}_{FG}, \mathcal{A}_{PG}, \mathcal{A}_{NG}$ be the ansatz designs of the circuits with parameters fully grouping, partially grouping, no-grouping, then the effective dimension related to $\mathcal{A}_{FG}, \mathcal{A}_{PG}, \mathcal{A}_{NG}$ yields*

$$d_{\text{eff}}(\mathcal{A}_{FG}) = d_{\text{eff}}(\mathcal{A}_{PG}) \leq d_{\text{eff}}(\mathcal{A}_{NG}), \tag{16}$$

where the equality in the inequality holds if there is no permutation symmetry in the problem Hamiltonian.

Proof of Theorem 3.1. To obtain the ordering relation of the convergence rate of various ansatz designs, we first elucidate the relation between the convergence rate of the approximation ratio and the effective dimension. Consider the problem Hamiltonian $H_C = \sum_{(i,j)} Z_i Z_j \in \mathbb{C}^{d \times d}$ with $d = 2^N$ and eigenvalues $\lambda_1 \leq \lambda_2 \cdots \leq \lambda_d$ and its corresponding eigenvector $\{|\lambda_i\rangle\}_{i=1}^d$. Preparing a quantum state $|\psi\rangle$ with overlap with the target ground state $|\psi^*\rangle$: $|\langle\psi|\psi^*\rangle| = 1 - \epsilon$, the lower bound of the expectation value of $\langle\psi|H_C|\psi\rangle$ is

$$\langle\psi|H_C|\psi\rangle = \langle\psi| \sum_{i=1}^{d} \lambda_i |\lambda_i\rangle \langle\lambda_i| \psi\rangle \tag{17}$$

$$= \lambda_1(1-\epsilon)^2 + \sum_{i=2}^{d} \lambda_i |\langle\psi|\lambda_i\rangle|^2 \tag{18}$$

$$\leq \lambda_1(1-\epsilon)^2 + \lambda_d(1-(1-\epsilon)^2), \tag{19}$$

where the first inequality works by scaling each eigenvalue λ_i to λ_d and following the fact $\sum_{i=2}^d |\langle\psi|\lambda_i\rangle|^2 \leq 1-(1-\epsilon)^2$. Then approximation ratio r is

$$\begin{aligned} r = \frac{\langle\psi|H_C|\psi\rangle}{\lambda_1} &\geq \frac{\lambda_d}{\lambda_1} - \frac{\lambda_d - \lambda_1}{\lambda_1}(1-\epsilon)^2 \geq (1-\varepsilon)^2, \\ \implies \quad \epsilon &\leq 1 - \sqrt{r} \end{aligned} \tag{20}$$

where the first inequality in the first equation holds because $\lambda_1 < 0$. Employing Lemma B.5, we have that the output state $|\psi^{(t)}\rangle$ converges to the solution state with approximation ratio $r \geq |\langle\psi^{(t)}|\psi^*\rangle|^2$ after $T_r = O(\log(\frac{d_{\text{eff}}}{1-\sqrt{r}}))$ iteration steps. These achieved results indicate that a small effective dimension leads to a faster convergence rate. In this regard, combining with Lemma B.6, the convergence rate T related to various ansatz $\mathcal{A}_{NG}, \mathcal{A}_{PG}, \mathcal{A}_{FG}$ for achieving the same approximation ratio yields $T_{FG} = T_{PG} \leq T_{NG}$. □

B.3 Proof of Lemma B.6

The proof of Lemma B.6 employs the following lemmas, where the proofs of Lemma B.7 and Lemma B.9 are deferred to Appendix B.4 and Appendix B.5.

Lemma B.7. *Let $\mathfrak{g}$ be a dynamical Lie algebra and r be the natural representation on the vector space V satisfying $r(g) = g$ for any $g \in \mathfrak{g}$. If there exists irreducible subrepresentations of r on V such that $r(g) = r_1(g) \oplus \cdots \oplus r_k(g)$ acting on the space $V = V_1 \oplus \cdots \oplus V_k$ for any $g \in \mathfrak{g}$, then the dimension of Lie algebra yields*

$$\dim(\mathfrak{g}) = \dim(r) = \sum_{j=1}^{k} \dim(r_j) = \sum_{j=1}^{k} \dim(V_j). \tag{21}$$

where the dimension of subrepresentation r_j refers to $\dim(r_j) = \dim(V_j)$.

Lemma B.8 (Commutant structure [38])**.** *Let r be a representation of a Lie algebra $\mathfrak{g}$ on the Hilbert space $\mathcal{H}$ and its decomposition into irreducible representation be*

$$r(g) = \oplus_{j=1}^{k} \mathbb{I}_{m_j} \otimes r_j(g), \tag{22}$$

where m_j is known as the multiplicity of the irreducible representation r_j. Then the elements of its commutant are of the following form

$$\mathcal{C}(\mathfrak{g}) = \oplus_{j=1}^{k} \mathcal{C}_j(\mathfrak{g}) \otimes \mathbb{I}_{\dim(m_j)}, \tag{23}$$

where $\mathcal{C}_j(\mathfrak{g})$ denotes bounded operators in a m_j-dimensional Hilbert space. Then the dimension of representation r and subrepresentation r_j yields

$$\dim(r) = \dim(\mathcal{C}(\mathfrak{g})), \textit{ and } \dim(r_j) = \dim(\mathcal{C}_j(\mathfrak{g})) \tag{24}$$

Lemma B.9. *Let $\mathfrak{g}_{FG}, \mathfrak{g}_{PG}, \mathfrak{g}_{NG}$ be the Lie algebra related to the ansatz designs of the circuits with parameters fully grouping $\mathcal{A}_{FG}$, partially grouping $\mathcal{A}_{PG}$, no-grouping $\mathcal{A}_{NG}$. Then the related commutants of the three Lie algebras yield*

$$\mathcal{C}(\mathfrak{g}_{NG}) \subseteq \mathcal{C}(\mathfrak{g}_{FG}) = \mathcal{C}(\mathfrak{g}_{PG}), \tag{25}$$

where the equality in the subset holds if there is no spatial symmetry in the problem Hamiltonian.

We now begin to present the proof of Lemma B.6.

Proof of Theorem B.6. Following Lemma B.7 with denoting r be the natural representation of $\mathfrak{g}$ on vector space V, the dimension of DLA $\mathfrak{g}$ is equal to the sum of dimensions of irreducible subrepresentations, i.e.,

$$\dim(\mathfrak{g}) = \dim(r) = \sum_{j=1}^{k} \dim(r_j) = \sum_{j=1}^{k} \dim(V_j), \tag{26}$$

where V_j is the irreducible invariant subspace related to the subrepresentation r_j. For the symmetric ansatz design $\mathcal{A}$, there exsits an invariant space $V_* \in \{V_j\}_{j=1}^{k}$ such that the effective dimension $d_{\text{eff}}(\mathcal{A}) = \dim(V_*)$.

To obtain Eqn. (16), we first show that the effective dimension of DLA $\mathfrak{g}$ is inversely proportional to the size of commutant of the DLA $\mathfrak{g}$, and then show that the commutant sizes related to ansatz design $\mathcal{A}_{FG}, \mathcal{A}_{PG}, \mathcal{A}_{NG}$ are monotonically non-increasing. In particular, the commutant of Lie algebra $\mathfrak{g}$, denoted as $\mathcal{C}(\mathfrak{g}) = \{V \in SU(d) : [V, g] = 0\}$, includes all the symmetry operator of the corresponding ansatz design. For any two Lie algebras $\mathfrak{g}_1, \mathfrak{g}_2$ with $\mathcal{C}(\mathfrak{g}_1) \subset \mathcal{C}(\mathfrak{g}_2)$, then any block diagonalization of the elements in $\mathcal{C}(\mathfrak{g}_1)$ is also the block diagonalization of the elements in $\mathcal{C}(\mathfrak{g}_2)$. This indicates that any invariant subspace of $\mathcal{C}(\mathfrak{g}_1)$ is also the invariant subspace of $\mathcal{C}(\mathfrak{g}_2)$, leading to $\dim(\mathcal{C}_j(\mathfrak{g}_2)) \leq \dim(\mathcal{C}_j(\mathfrak{g}_1))$. Following Lemma B.8, we have

$$d_{\text{eff}}(\mathfrak{g}_2) = \dim(r_*(\mathfrak{g}_2)) = \dim(\mathcal{C}_*(\mathfrak{g}_2)) \leq \dim(\mathcal{C}_*(\mathfrak{g}_1)) = \dim(r_*(\mathfrak{g}_1)) = d_{\text{eff}}(\mathfrak{g}_1), \tag{27}$$

where $* \in [k]$ refers to the index of invariant space the optimization performs on and $\dim(r_*(\mathfrak{g}_j))$ with $j = 1, 2$ refers to the effective dimension related to the DLA $\mathfrak{g}_j$. In conjunction with Lemma B.9 and Eqn. (27), we have $\mathcal{C}(\mathfrak{g}_{NG}) \subseteq \mathcal{C}(\mathfrak{g}_{PG}) = \mathcal{C}(\mathfrak{g}_{FG})$ and hence $d_{\text{eff}}(\mathfrak{g}_{FG}) = d_{\text{eff}}(\mathfrak{g}_{PG}) \leq d_{\text{eff}}(\mathfrak{g}_{NG})$. This completes the proof. □

B.4 Proof of Lemma B.7

Proof of Lemma B.7. The first equality in Eqn. (21) follows the fact that natural representation r is bijective and does not change the dimension of pre-image space. the second equality follows the definition of the dimension of representation in Definition B.2 such that

$$\dim(r) = \dim(V) = \dim(V_1 \oplus \cdots \oplus V_k) = \sum_{j=1}^{k} \dim(V_j) = \sum_{j=1}^{k} \dim(r_j), \tag{28}$$

where the last equality follows that r_j is a representation of $\mathfrak{g}$ on the space V_j. This completes the proof. □

https://doi.org/10.52202/079017-1061

B.5 Proof of Lemma B.9

Proof of Lemma B.9. We begin this proof by showing that the commutant of $A = \{H_1 \otimes \mathbb{I}, \mathbb{I} \otimes H_2\}$ is a subset of $B = \{H_1 \otimes \mathbb{I} + \mathbb{I} \otimes H_2\}$, where H_1, H_2 are arbitrary Hermitian operators and B refers to the set with imposing parameter grouping on A. In particular, for any matrix S which commutes with the elements in A, we have

$$S(H_1 \otimes \mathbb{I} + \mathbb{I} \otimes H_2) = S(H_1 \otimes \mathbb{I}) + S(\mathbb{I} \otimes H_2) = (H_1 \otimes \mathbb{I})S + (\mathbb{I} \otimes H_2)S = (H_1 \otimes \mathbb{I} + \mathbb{I} \otimes H_2)S. \quad (29)$$

This indicates that $\mathcal{C}(A) \subseteq \mathcal{C}(B)$. With this fact, we now derive the Eqn. (25). We first recall that the generators of the Lie algebras $\mathfrak{g}_{NG}, \mathfrak{g}_{PG}, \mathfrak{g}_{FG}$ yield non-discreasingly restrictive parameters grouping strategy, and are identity when there is no spatial symmetry in the problem Hamiltonian, i.e., $\mathfrak{g}_{FG} = \mathfrak{g}_{PG} = \mathfrak{g}_{NG}$. Moreover, the definition of $\mathfrak{g}_{FG}, \mathfrak{g}_{PG}$ indicates that the related ansatzes follow the same symmetry, namely, any unitary U commutes with the elements in $\mathfrak{g}_{FG}$ if and only if U commutes with the elements in $\mathfrak{g}_{PG}$. Hence we have $\mathcal{C}(\mathfrak{g}_{FG}) = \mathcal{C}(\mathfrak{g}_{PG})$ as the commutant consists of the symmetry operator of the ansatz design.

On the other hand, the relation $\mathcal{C}(\mathfrak{g}_{PG}) \subseteq \mathcal{C}(\mathfrak{g}_{NG})$ in Eqn. (25) directly following the analog between the set A and B and the generators related to the Lie algebra $\mathfrak{g}_{PG}$ and $\mathfrak{g}_{NG}$, where the generators related to $\mathfrak{g}_{PG}$ refers to the set with imposing parameters grouping on $\mathfrak{g}_{NG}$. This completes the proof. □

C Related work

In this section, we embark on a concise literature review, focusing on conventional algorithms for the Max-Cut problem, some variants of QAOA, and quantum circuit architecture search algorithms. This examination sets the stage for a comparative analysis between these established methods and our proposed model. In summary, our discussion underscores the distinctive strength of our model: its exceptional ability to generalize.

C.1 Conventional algorithms

Greedy algorithm for Max-Cut problem. The greedy algorithm for solving the Max-Cut problem operates on a simple principle: iteratively makes local, myopic decisions to construct a solution that attempts to maximize the sum of weights of edges between two disjoint subsets of vertices. This algorithm does not assure an optimal solution due to its greedy nature—making decisions based only on immediate benefits without considering future consequences. The detailed procedure is introduced in Alg. 1.

Goemans-Williamson (GW) algorithm for Max-Cut problem. The GW algorithm utilizes semidefinite programming to relax the original combinatorial problem into a continuous one that can be solved efficiently. After solving the semidefinite program, the algorithm uses a random hyperplane to split the vertices into two subsets, which form the cut. The GW algorithm achieves an approximation ratio of at least 0.878 for the Max-Cut problem. The simplified pseudocode of GW algorithm is described in Alg. 2.

C.2 Variants of QAOA

The studies of variants of QAOA aim to improve the convergence rate or reduce the computational time by changing the PQCs or the problem Hamiltonian. Current progress has revealed that the performance of QAOA could be improved by employing multi-angle QAOA [17] where the parameters are no-grouped or partially grouped according to the permutation symmetry of problem Hamiltonian [39, 40, 23], utilizing different mixer Hamiltonian obtained by searching from a given Hamiltonian pool [34] or inspired by specific problem [41, 22, 42, 43] and other quantum algorithms [33, 44, 45]. Another type of the variant of QAOA focuses on modifying the problem Hamiltonian, either through eliminating redundant qubits [46] to obtain a reduced problem Hamiltonian, or imposing conditional rotations [47] to the Hamiltonian. In the following, we delve into the most relevant variants of QAOA to our study and compare them with our model.

Multi-Angle QAOA (ma-QAOA) [17]. The ma-QAOA innovates on the traditional QAOA framework by incorporating a larger set of parameters. It allows each operator within both the cost and

https://doi.org/10.52202/079017-1061

Algorithm 1 Greedy Algorithm for weighted Max-Cut

```
Input: A graph $G = (V, E)$ with weights $w_{ij}$ on edges $(i, j) \in E$
Output: A partition of $V$ into subsets $S$ and $\bar{S}$ maximizing the cut weight
Initialize $S = \emptyset$, $\bar{S} = V$
Initialize $cutWeight = 0$
for each vertex $v \in V$ do
    $deltaWeight = 0$
    for each edge $(v, u) \in E$ connected to $v$ do
        if $u \in S$ and $v \notin S$ or $u \notin S$ and $v \in S$ then
            $deltaWeight = deltaWeight - w(v, u)$
        else
            $deltaWeight = deltaWeight + w(v, u)$
        end if
    end for
    if $deltaWeight > 0$ then
        if $v \in S$ then
            Move $v$ to $\bar{S}$ and update $cutWeight+ = deltaWeight$
        else
            Move $v$ to $S$ and update $cutWeight+ = deltaWeight$
        end if
    end if
end for
return $S$, $\bar{S}$, $cutWeight$
```

Algorithm 2 Goemans-Williamson Algorithm for Max-Cut

```
Input: A graph $G = (V, E)$ with weights $w_{ij}$ on edges $(i, j) \in E$
Output: A partition of $V$ into subsets $S$ and $\bar{S}$
Formulate the Max-Cut problem as a semidefinite programming (SDP) problem.
Solve the SDP problem to find a vector representation $\vec{v}_i$ for each vertex $i$.
Choose a random hyperplane by selecting a random unit vector $\vec{r}$.
for each vertex $i \in V$ do
    if $\vec{v}_i \cdot \vec{r} \geq 0$ then
        Assign vertex $i$ to subset $S$
    else
        Assign vertex $i$ to subset $\bar{S}$
    end if
end for
return $S$, $\bar{S}$
```

mixer Hamiltonians to be governed by its own unique parameter, diverging from the conventional approach where a single parameter is shared among all operators. In our experiment, attention is focused exclusively on the modifications within the mixer Hamiltonian for fair comparison. The new mixer Hamiltonian is expressed as

$$H_M = \sum_{i=1}^{N} \beta_i X_i. \tag{30}$$

where X_i denotes the Pauli-X operation applied to the i-th qubit and β_i represents the corresponding individual parameter. This adjustment significantly expands the parameter space in ma-QAOA, scaling the total count from $2p$ in the standard QAOA to $(N + 1)p$. Despite empirical evidence suggesting that ma-QAOA surpasses the original QAOA in achieving higher approximation ratios for configurations with fewer layers, the complexity introduced by the augmented parameter space could potentially impede its effectiveness in scenarios involving deeper circuits.

ADAPT-QAOA [34]. In ADAPT-QAOA, the mixer Hamiltonian is selected from a pre-defined operator pool $\{A_j\}$ step by step. For the k-step, the operators A_j is guided by maximizing the following gradient:

$$-i \langle \psi_{k-1}(\boldsymbol{\alpha}, \boldsymbol{\beta}) | e^{i\alpha_k H_C [H_C, A_j] e^{-i\alpha_k H_C} | \psi_{k-1}(\boldsymbol{\alpha}, \boldsymbol{\beta})} \rangle, \tag{31}$$

where $|\psi_p(\boldsymbol{\alpha}, \boldsymbol{\beta})\rangle = \left(\prod_{k=1}^{p} e^{-i\beta_k A_k} e^{-i\alpha_k H_C}\right)|\psi_0\rangle$. Following the selection of A_j, all parameters undergo a subsequent optimization phase. This procedure is iterated until the gradient's norm falls below a set threshold, or the circuit reaches its predefined maximum depth. ADAPT-QAOA's dynamic mixer Hamiltonian selection aims to potentially discover a more direct path to adiabaticity, thereby enabling accelerated convergence. However, its practicality for large-scale problems is hampered by the increased measurement costs required for gradient evaluation, a factor contingent on the size of the operator pool.

Contrasting with these QAOA variants, MG-Net uniquely offers a dynamic offline adaptation of the mixer Hamiltonian, tailoring it to the specific problem and circuit depth without incurring extra computational costs. Additionally, MG-Net demonstrates remarkable generalization capabilities, effectively learning from a limited dataset to address a broad spectrum of problems. This facilitates the rapid development of mixer Hamiltonians for new problems.

C.3 Quantum circuit architecture search

In the design of quantum circuits, quantum circuit architecture search methodologies have been developed to autonomously identify optimal quantum circuit architectures [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59]. In the following, we delve into several notable approaches and contrast them with our MG-Net model.

Quantum architecture search (QAS) [53]. The QAS approach automatically seeks an optimal quantum circuit architecture to balance the benefits and side effects of adding more quantum gates, considering the noise in quantum systems. This method involves several steps: initializing a superstructure (supernet) that defines the pool of potential architectures, optimizing parameters across these architectures, ranking them based on performance, and finally refining the chosen architecture.

Differentiable Quantum Architecture Search (DQAS) [56]. DQAS introduces a novel approach by employing differentiable programming techniques. This method enables the concurrent optimization of both the structure and parameters of quantum circuits through gradient descent, streamlining the search process.

QuantumDARTS [57]. The QuantumDARTS algorithm, which leverages the Gumbel-Softmax technique for differential optimization of quantum circuit structure and parameters, aims to reduce the search cost by following two search strategies: macro search for entire circuit optimization and micro search for sub-circuit structures, improving its adaptability to large-scale problems.

Despite their advancements, these QAS methodologies share a fundamental limitation: they are inherently designed to address singular, specific problems. Consequently, adapting these methods to new problems necessitates repeating the resource-intensive architecture search process from scratch. In contrast, MG-Net exhibits an unparalleled ability to generalize across a spectrum of problems based on a minimal set of training examples. This capability enables MG-Net to rapidly design optimal circuits for novel problems through a single feedforward computation, bypassing the need for repeated, exhaustive searches. This unique advantage positions MG-Net as a highly efficient and versatile tool in the quantum computing landscape, offering significant savings in computational resources and time.

D Implementation details of MG-Net

In this section, we initially outline the methodology for constructing datasets used to train MG-Net across various problem scales. Subsequently, we detail the implementation of the data encoder, illustrated with a specific example.

D.1 Dataset construction

Operator types. The set of operator types for the mixer Hamiltonian is defined as $\{X, Y\}^{\otimes N}$ in our experiments. Note that the operator type pool can be flexibly adjusted according to specific problems and hardware. For example, we can introduce two-qubit operators into the operator type pool to further enhance the performance of QAOA, as done in [60]. Considering the exponential growth of the search space in relation to the system size N, we have sampled only a subset from this pool in all

https://doi.org/10.52202/079017-1061

our experiments. This approach is adopted to construct the training dataset while minimizing data collection costs.

Construction of parameter group pool. A straightforward idea to construct the pool of parameter group is to assume each X_i can be assigned an index j ranging from 1 to N, leading to a pool $P = \{(j_1 \in [N], ..., j_N \in [N])\}$ with size N^N. However, there exist multiple duplicate candidates in the pool P due to the disorder of the initial parameter pool. For example, for a two-qubit QAOA ansatz, parameter index vectors $(1, 2)$ and $(2, 1)$ make no difference in the optimization of QAOA. Based on these observations, we propose a recursive algorithm Alg. 3 to build a compact pool of parameter groups.

Algorithm 3 **Construction of parameter group pool**

```
Input: The qubit number $N$, pool $P = \{\}$
Output: Pool $P$
Function grouping_pool($max\_index, index\_list, N$)
    if length($index\_list$) $==$ $N$
        Add $index\_list$ to $P$
        return
    end if
    for $i = 1, \cdots, max\_index$
        Append $i$ to $index\_list$
        grouping_pool($\max(max\_index, index\_list[-1] + 2), index\_list, N$)
        Delete the last element of $index\_list$
    end for
End Function
grouping_pool($2$, empty_list, $N$)
```

In practice, we randomly selected 5 candidates from the parameter grouping pool for each operator type. Although the training dataset only partially covers the entire space of operator types and parameter groupings, our model is still capable of learning the intrinsic relationship between the mixer Hamiltonian and its corresponding achievable cost.

To find the minimal cost that can be achieved by a QAOA circuit during the construction of the training dataset in stage 1, we run the same QAOA circuit 10 times and record their cost values. For each run, the QAOA circuit is initialized with different random parameters and optimized for 40 epochs. Finally, the minimum of these cost values is selected as the label that represents the minimal achievable cost.

Large-scale dataset. To assess our method's efficacy on large-scale problems, we concentrated on the Max-Cut problem using weighted graphs with 64 nodes. Simulating larger-scale quantum circuits on classical devices poses significant challenges. To overcome this, our approach employs a divide-and-conquer strategy, simulating a large-scale circuit through multiple smaller-scale circuits. We then integrate the results of these smaller circuits to estimate the performance of the original large-scale circuit. For a detailed explanation of this methodology, refer to QAOA-in-QAOA [61].

In constructing the training dataset $D_{\text{ce}}^{\text{Tr}}$ for 64-node graphs, we divide each 64-node graph into 8 sub-graphs, each containing 8 nodes. The max-cut of each sub-graph is computed using an 8-qubit QAOA. To gather a comprehensive range of samples, we vary the operator types and parameter groupings in the 8-qubit circuits, which in turn simulates the variation in mixer Hamiltonians for 64-qubit circuits. It is important to note that these 8-qubit circuits operate independently, with no shared parameters, resulting in at least 8 independent parameters for each 64-qubit circuit in our training dataset. For testing on the unknown graphs, we employ tensor network simulations to accurately estimate the performance of the original 64-qubit QAOA.

D.2 Data encoder

Problem encoder. Our problem encoder is rooted on the problem Hamiltonian H_C in Eqn. (1). More precisely, to facilitate a consistent and unified representation for diverse combinatorial problems $\{G\}$, we initiate by converting the original problem G into the corresponding unitary $U_C = \exp(-i\alpha_k H_C)$, which is subsequently transformed into a directed acyclic graph (DAG) G_C.

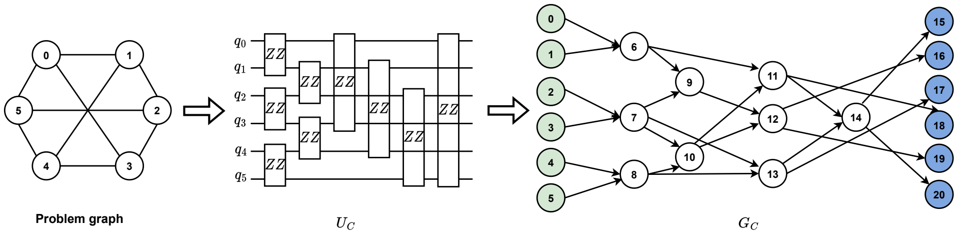

Figure 6: **Encoding of problem.** The problem graph is first transformed into a quantum circuit, which is subsequently encoded by a DAG.

Fig. 6 illustrates the problem encoding process for a regular graph with 6 nodes. Each node of the problem graph corresponds to a qubit in the quantum system and each edge (i, j) is represented as a two-qubit gate $Z_i Z_j$, which is exactly the problem Hamiltonian of QAOA for the Max-Cut problem. Based on this problem unitary, we construct the final graph representation G_C, with each two-qubit gate depicted as a node in the graph. In addition to these gate-induced nodes, two unique node types, the input and output nodes which correspond to qubits, are introduced to denote the start and end of G_C, respectively. The edges of G_C signify the temporal order of quantum gate execution, linking consecutive gates and thereby dictating the flow of the quantum computation. The weights of edges are encoded into the node feature.

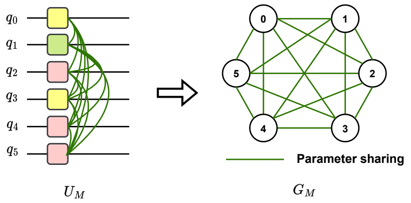

Figure 7: **Encoding of mixer Hamiltonian.** Each qubit in the mixer Hamiltonian is represented as a node in the encoded graph. The type of operator associated with each qubit is encoded in the node feature, while the parameter grouping strategy is encapsulated in the edge features.

Mixer encoder. We define a one-to-one mapping to encode the candidate mixer Hamiltonian H_M as a graph G_M. Recall Eqn. (5), two types of information about H_M should be encoded in G_M are operators $\{P_i\}$ and the parameter grouping strategy $\mathcal{G}$. In MG-Net, each operator is modeled as a node of G_M, and the operator type is encoded as part of the node feature vector. Concretely, MG-Net initially constructs G_M as a fully connected graph, where the edge weight is a binary variable, representing whether the two operators connected by the edge share the same control parameter.

The process of encoding a mixer Hamiltonian into a graph representation is illustrated in Fig. 7. Here, we take the example of a 6-qubit mixer Hamiltonian encoded as graph G_M. In this graph, each qubit's corresponding operator is depicted as a node, with the operator acting on the i-th qubit represented by the i-th node in G_M. The graph's edges signify the parameter correlations among these operators. Specifically, let $w_{ij} \in \{0, 1\}$ be the weight of edge connecting node i and j. If the operator i and j share the same parameter, then $w_{ij} = 0$; otherwise, $w_{ij} = 1$.

Depth embedding. The circuit depth p is encoded as a vector $\boldsymbol{x}_p$ through position embedding [62]. Mathematically, $\boldsymbol{x}_p$ is constructed as

$$\boldsymbol{x}_p[2k] = \sin \frac{p}{10000^{2k/d_p}}, \boldsymbol{x}_p[2k+1] = \cos \frac{p}{10000^{2k/d_p}},$$

where d_p is dimension of $\boldsymbol{x}_p$ and $k = 0, ..., \lfloor d_p/2 \rfloor$.

D.3 Network structure

D.3.1 Cost estimator

In our experimental setup, the intricate architecture of the cost estimator is detailed in Fig. 8. Both the problem and mixer Hamiltonian branches incorporate two layers of graph convolutions, utilizing

ReLU activation functions to transform the initial node features from dimensions d_C and d_M to a unified 128-dimensional space. Subsequently, the three extracted features—$\boldsymbol{x}_C$, $\boldsymbol{x}_M$, and $\boldsymbol{x}_p$—are concatenated to facilitate the prediction of the attainable minimum cost $\hat{y}$ for a given QAOA instance through an MLP layer.

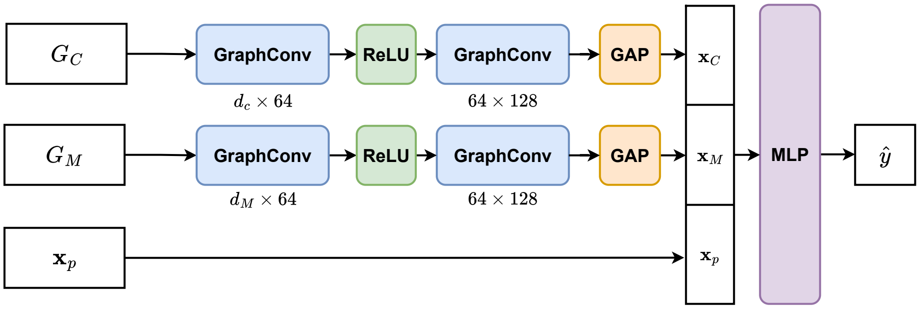

Figure 8: **Implementation of cost estimator.** The term 'GraphConv' represents the graph convolution module. 'ReLU' is a commonly used activation function in neural networks. d_C and d_M represent the dimension of node feature in graph G_C and G_M respectively. P_i represents the operator type for the i-qubit and $\boldsymbol{e}_{ij}$ represents the weight for edge (i, j).

D.3.2 Mixer generator

Inspired by [63] which encodes a quantum circuit as a graph, the mixer generation is composed of two separate sub-generators: the operator type generator and the parameter grouping generator, which are respectively responsible for graph node and link prediction.

Operator type generator. The task of generating operator types $\mathcal{P}$ is conceptualized as a graph node classification task. Specifically, we employ a GNN to process G_C, identifying output nodes to represent the operators corresponding to each qubit, while disregarding irrelevant nodes. To incorporate the circuit depth p into the prediction, we enhance the feature set of each output node by appending a feature vector $\boldsymbol{x}_p$. This enriched node feature set is then fed into an MLP to predict the specific category of each operator.

Parameter grouping generator. Recall that the grouping strategy is traditionally represented by sets of index groups $\{\mathcal{G}_j\}_{j=1}^K$ with an unspecified K, posing a challenge for neural network processing. To address this, we extend the parameter grouping problem as follows: if an edge indicator $\boldsymbol{e}_{ij} = 1$, then the mixer operators P_i and P_j are correlated and share the same parameter; otherwise, they are controlled by independent parameters. Furthermore, if $\boldsymbol{e}_{ij} = 1$ and $\boldsymbol{e}_{ik,k\neq j} = 1$, then P_i, P_j and P_k are correlated regardless of the value of $\boldsymbol{e}_{jk}$. In this way, the parameter grouping task is translated into the prediction of the binary variable $\boldsymbol{e}_{ij} \in \{0, 1\}$, as a link prediction task. This modeling bypasses the need to predetermine the number of parameter groups and offers flexibility in incorporating constraints related to qubit connections.

Analogous to the operator type generator, the parameter grouping generator employs another GNN to process G_C to extract features of output nodes, which are then extended with circuit depth feature $\boldsymbol{x}_p$. For node i and j, their extended features $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are used to determine the existence of an edge (i, j) by evaluating $\boldsymbol{e}_{ij} = B(\mathrm{MLP}(\boldsymbol{x}_i \circ \boldsymbol{x}_j))$, where $B(\cdot)$ signifies a binarization function. In MG-Net, this function is realized using the Gumbel-Softmax trick, ensuring the differentiability.

In our experiment, the detailed structure of the mixer generator is depicted in Fig. 9. The mixer generator integrates two specialized branches to analyze the input problem graph G_C, with each branch deploying two graph convolution layers to distill the feature vector $\boldsymbol{x}_C$ with a dimensionality of 128. This feature vector is then augmented with the circuit depth feature $\boldsymbol{x}_p$ to enrich the predictive capability of the model. For the precise prediction of operator types $\{P_i\}_{i=1}^N$ applicable to each qubit, the terminal nodes of G_C are chosen for input into a Multi-Layer Perceptron (MLP) layer. This step calculates the likelihood of each potential operator type. Concurrently, a separate MLP layer is employed to ascertain the parameter sharing between operators P_i and P_j. This is achieved through the equation $eij = \mathrm{MLP}(\boldsymbol{x}_i \circ \boldsymbol{x}_j)$, where $\circ$ denotes the element-wise multiplication, and $\boldsymbol{x}_i$ symbolizes the enriched feature of the i-th node.

https://doi.org/10.52202/079017-1061

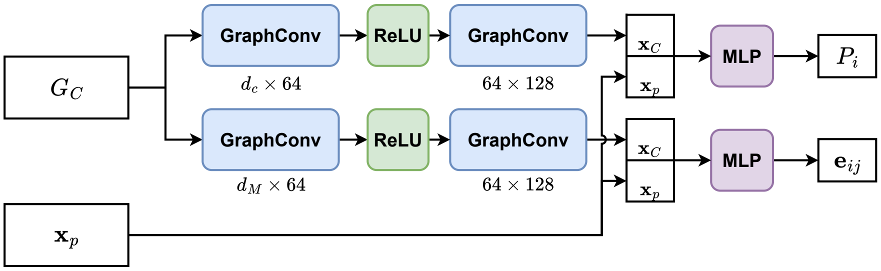

Figure 9: **Implementation of mixer generator.** The term 'GraphConv' represents the graph convolution module. 'ReLU' is a commonly used activation function in neural networks. d_C and d_M represent the dimension of node feature in graph G_C and G_M respectively.

D.4 Experiment settings

Hardware platform. All QAOA circuits are implemented by PennyLane [64] and run on classical device with Intel(R) Xeon(R) Gold 6267C CPU @ 2.60GHz and 128 GB memory. MG-Net is implemented by Pytorch [65] and is trained on a single NVIDIA GeForce RT 2080Ti with 12G graphics memory.

Hyper-parameters. The hyper-parameters of optimizing MG-Net and QAOA circuit are listed in Tab. 3.

Initial state. The initial quantum state of the QAOA circuit is consistently set to $|+\rangle^{\otimes N}$, irrespective of the mixer Hamiltonian chosen. Although this approach does not ensure that the initial state is always the ground state of the predicted mixer Hamiltonian, it does not compromise the QAOA's performance and has the potential to outperform the traditional state initialization technique, which can be partially explained by the physical intuition of counterdiabatic (CD) driving [33, 34].

Table 3: **The hyper-parameters of optimizing MG-Net and QAOA circuit.**

	QAOA	MG-Net
optimizer	Adam	Adam
learning rate	0.15	$1 * 10^{-4}$
epoch	40	250
λ_e	-	1.0
λ_r	-	1.0

E More numerical results

In this section, we initially show the results of comparing the approximation ratio achieved by different methods for TFIM. Then we examine how the approximation ratio achieved by various methods varies with different circuit depths p. Subsequently, we explore the convergence behavior of the QAOA when enhanced by our approach.

E.1 Performance comparison among different methods for TFIM

In evaluating the effectiveness of our proposed method for solving TFIM, we conducted a comparative analysis against QAOA, ADAPT-QAOA, and multi-angle QAOA (ma-QAOA). Our analysis, based on the average results from 100 graphs in our test dataset, is summarized in Tab. 4. The findings reveal that our method consistently outperforms other techniques in achieving a higher approximation ratio for TFIM, particularly in larger-scale problems.

E.2 Experiments on asymmetric graphs and 2D-TFIM

We conducted additional experiments on the asymmetric graphs of 6 nodes and 2D lattice models of 6 spins. Their topological structure is shown in Fig. 10.

https://doi.org/10.52202/079017-1061

Table 4: **Comparison of approximation ratio r among different methods for TFIM.**

Method	6 qubits	16 qubits
QAOA	0.990 ± 0.005	0.523 ± 0.083
ADAPT-QAOA	0.857 ± 0.245	0.742 ± 0.356
ma-QAOA	0.994 ± 0.001	0.921 ± 0.040
Ours	$\mathbf{0.996 \pm 0.001}$	$\mathbf{0.963 \pm 0.031}$

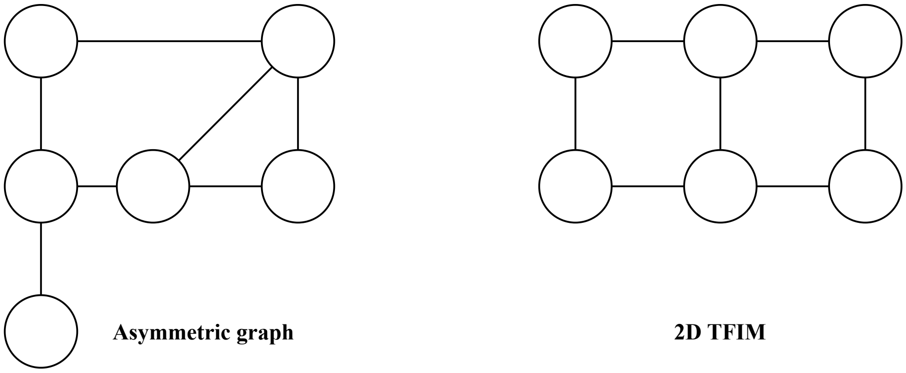

Figure 10: **Topological structure of asymmetric graphs and 2D TFIM.**

The comparison of the achieved approximation ratio at $p = 42$ over 100 random test samples is summarized in the Tab. E.2. The result affirms that our model consistently outperforms both standard QAOA and ma-QAOA in terms of approximation ratio on more general cases.

Tasks	Max-Cut for asymmetric graphs	2D TFIM
QAOA	0.952 ± 0.026	0.977 ± 0.008
ma-QAOA	0.987 ± 0.008	0.980 ± 0.019
Ours	$\mathbf{0.988 \pm 0.005}$	$\mathbf{0.988 \pm 0.006}$

E.3 Approximation ratio with respect to p

In small-scale quantum systems, achieving the criteria set in Theorem 3.1 is more straightforward by increasing circuit depth p beyond the threshold C. We analyze the approximation ratios achieved by 6-qubit QAOA circuits for Max-Cut and TFIM within the p range of 2 to 82. Figure 11 illustrates that at lower p values, our method consistently records the highest approximation ratio r, clearly outperforming both standard QAOA and ma-QAOA. As p increases from 2 to 62, standard QAOA and ma-QAOA exhibit a rise in r, eventually matching our method's performance. However, a further increase in p leads to a performance decline in ma-QAOA, where the detrimental impact of its numerous trainable parameters on convergence outweighs the benefits of enhanced expressibility. In contrast, our method maintains stable performance, continually achieving the highest r. These findings confirm our method's superiority in optimizing approximation ratios across various circuit depths compared to other approaches.

We further explore the specific configurations of mixer Hamiltonians generated by MG-Net. Table 5 presents examples of predicted mixer Hamiltonians for p values of $12, 52, 82$. At a smaller circuit depth of $p = 12$, the optimal parameter grouping strategy maximizes the number of parameters, assigning each operator its independent parameter. This approach enhances the expressivity of the QAOA circuit and, alongside the introduction of novel mixer operators, contributes to superior approximation performance. For $p = 52$, which verges on the threshold of over-parameterization, a trend towards grouping some operators is observed. At a higher circuit depth, such as $p = 82$, the majority of operators are assigned the same parameter, aligning closer to the configuration of

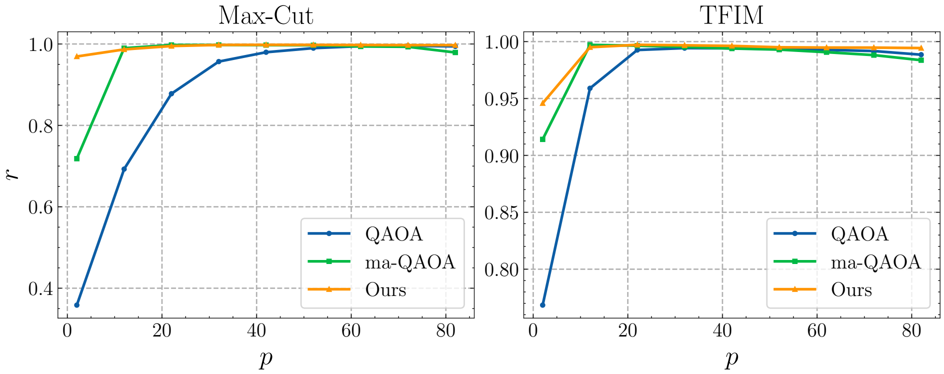

Figure 11: **Comparison of the approximation ratio achieved by 6-qubit QAOA, ma-QAOA and our model for Max-Cut and TFIM with varying p.**

a standard QAOA circuit. The evolution of the mixer Hamiltonian configuration with varying p partially reveals the underlying design principle of mixer Hamiltonian across different problems and circuit depths.

Table 5: **Operator type and parameter group generated by MG-Net.** 'X' and 'Y' represent Pauli-X and Pauli-Y, respectively. Parameter groups are formatted as $a_1 - a_2 - \cdots - a_N$, with $a_i \in \{0, 1, ..., N-1\}$ indicating the parameter index for the i-th operator. Identical indices ($a_i = a_j$) imply shared parameters between operators.

Task		Max-Cut	TFIM
$p = 12$	Operator type	YYYYXX	XXXXXX
	Parameter Group	0-1-2-3-4-5	0-1-2-3-4-5
$p = 52$	Operator type	XXXXXX	XXXXXX
	Parameter Group	0-1-2-0-4-4	0-1-1-3-4-5
$p = 82$	Operator type	XXXXXX	XXXXXX
	Parameter Group	0-1-0-0-0-1	0-0-0-0-0-0

E.4 Convergence of QAOA with various mixer Hamiltonian

In our investigation, we conducted an analysis on a randomly selected 16-qubit Max-Cut and TFIM problem from our test dataset, scrutinizing the convergence patterns of QAOA, ma-QAOA, and our method across various configurations ($p = 4, 6, 8, 10$). Illustrated in Fig. 12, our methodology not only achieves a notably lower loss value within a reduced number of iterations in comparison to both QAOA and ma-QAOA but also consistently outperforms in terms of the final loss value attained by the end of the optimization. Specifically, at $p = 10$, our approach necessitates merely 28 iterations for Max-Cut and 22 iterations for TFIM to diminish the loss value to -8 and -15, respectively. In contrast, ma-QAOA demands 40 iterations for both challenges, whereas QAOA fails to achieve this loss value. This evidence underscores the superior efficiency and effectiveness of our method in navigating the solution landscape for these quantum optimization tasks.

E.5 Experiments on extended candidate operator type set

In this section, we investigate the performance of our model when applied to a more complex set of candidate operator types. Specifically, we expand the pool of mixer operator types from X, Y to X, Y, XX, YY by incorporating additional two-qubit operators, thereby increasing the search space for operator types to $O(4^N)$. All other experimental conditions remain consistent with those described in the main text. The behavior of the cost estimator under these conditions is illustrated in Fig. 13. Our results indicate that the cost estimator continues to serve as a reliable performance indicator for QAOA, even with the increased complexity of the mixer Hamiltonian design.

https://doi.org/10.52202/079017-1061

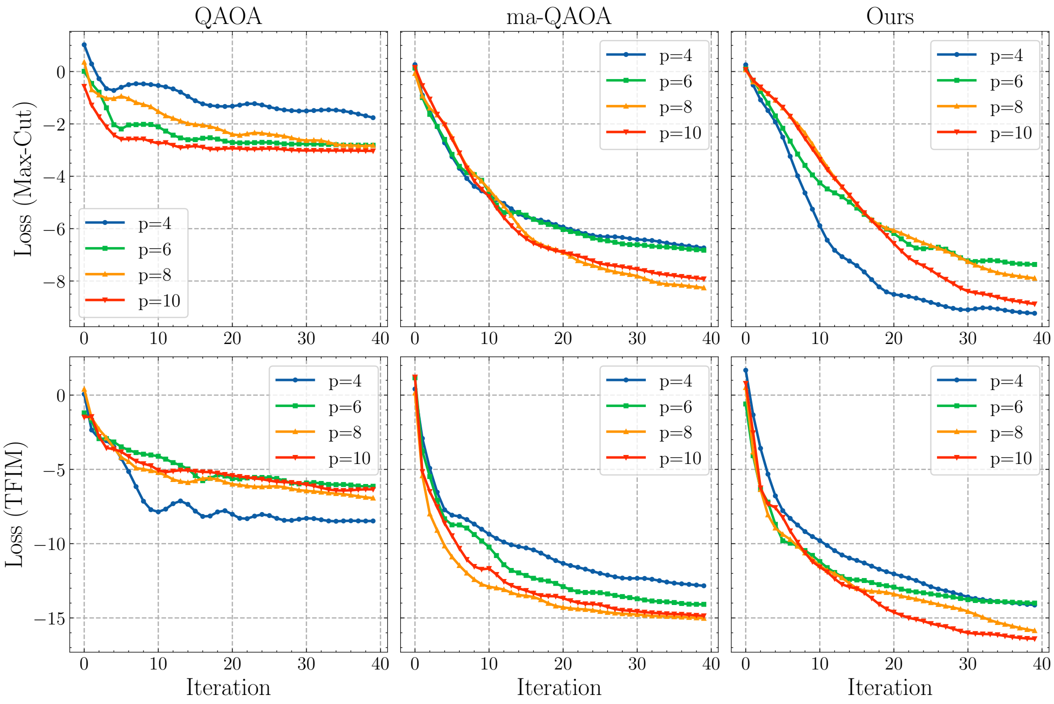

Figure 12: **Comparison of the convergence of 16-qubit QAOA, ma-QAOA and our model for Max-Cut and TFIM with varying p.**

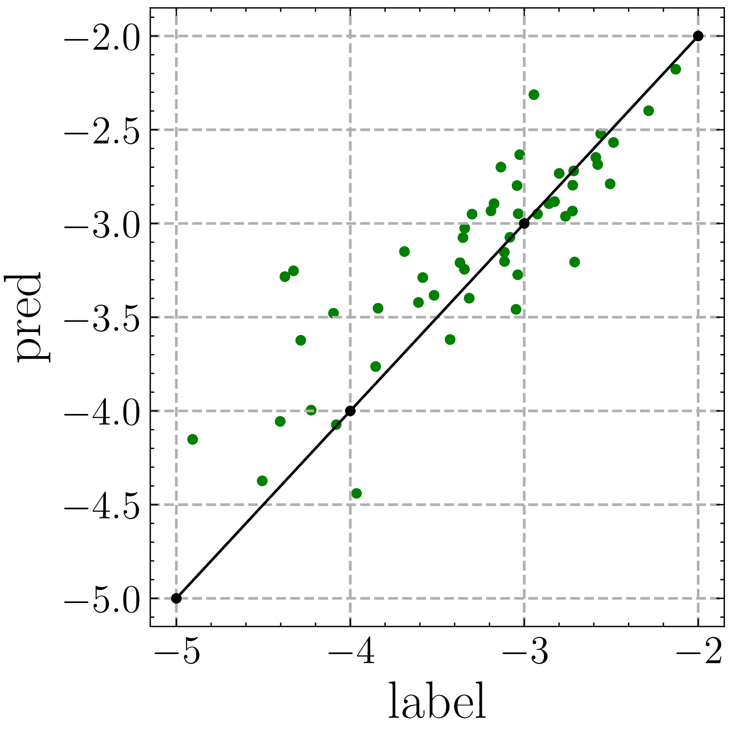

Figure 13: **Behavior of cost estimator with extended mixer operator pool $\{X, Y, XX, YY\}$.** 'label' represents the actual achieved approximation ratio, while 'pred' represents the result predicted by the cost estimator.

https://doi.org/10.52202/079017-1061

E.6 Ablation study on the circuit depth embedding

MG-Net acts as an initial protocol and provides a flexible circuit-generation framework where model components can be conveniently replaced by advanced techniques. Besides the position embedding of circuit depth in the main text, we have also considered another two embedding strategies: integer embedding and one-hot embedding. There are two key differences between the implementation of position encoding and one-hot or integer encoding:

1. **Feature vector length.** The length of the one-hot-encoded vector $\mathbf{x}_p$ depends on the predefined maximum value of p, while the length of the integer-encoded vector $\mathbf{x}_p$ is 1. In contrast, we adjust the length of position-encoded vector $\mathbf{x}_p$ according to the dimension of $\mathbf{x}_C$ and $\mathbf{x}_M$.
2. **Feature integration strategy.** When using one-hot or integer encoding, we employ concatenation as the integration strategy for the three features $\mathbf{x}_C$, $\mathbf{x}_M$ and $\mathbf{x}_p$ rather than summation.

The achieved approximation ratios for 6-qubit MaxCut problems using different depth encoding methods are shown below:

Table 6: **Comparison of approximation ratio r among different circuit depth embedding strategies.**

Depth embedding method	Approximation ratio r
Integer	0.981 ± 0.004
One-hot	0.984 ± 0.003
Position	0.99 ± 0.0004

https://doi.org/10.52202/079017-1061

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Sec. 1

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Sec. 6

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

https://doi.org/10.52202/079017-1061

Justification: Sec. 3

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Sec. 1 and Sec. 5

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

https://doi.org/10.52202/079017-1061

Answer: [Yes]

Justification: Sec. 1

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 5

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Sec. 5

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

https://doi.org/10.52202/079017-1061

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix. D

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Sec. 6

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

https://doi.org/10.52202/079017-1061

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Appendix. D

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

https://doi.org/10.52202/079017-1061

Answer: [Yes]

Justification: `https://github.com/QQQYang/MG-Net`

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-1061