PAPER

Differentiable quantum architecture search

Quantum Science and Technology

RECEIVED
8 May 2022

REVISED
31 July 2022

ACCEPTED FOR PUBLICATION
8 August 2022

PUBLISHED
24 August 2022

PAPER

Differentiable quantum architecture search

Shi-Xin Zhang[1,2], Chang-Yu Hsieh[2,*], Shengyu Zhang[2] and Hong Yao[1,3,*]

[1] Institute for Advanced Study, TsinghuaUniversity, Beijing 100084, People's Republic of China
[2] Tencent Quantum Laboratory, Tencent, Shenzhen, Guangdong 518057, People's Republic of China
[3] State Key Laboratory of Low Dimensional Quantum Physics, Tsinghua University, Beijing 100084, People's Republic of China
[*] Authors to whom any correspondence should be addressed.

E-mail: kimhsieh2@gmail.com and yaohong@tsinghua.edu.cn

Keywords: variational quantum algorithms, quantum architecture search, quantum simulations

Supplementary material for this article is available online

Abstract

Quantum architecture search (QAS) is the process of automating architecture engineering of quantum circuits. It has been desired to construct a powerful and general QAS platform which can significantly accelerate current efforts to identify quantum advantages of error-prone and depth-limited quantum circuits in the NISQ era. Hereby, we propose a general framework of differentiable quantum architecture search (DQAS), which enables automated designs of quantum circuits in an end-to-end differentiable fashion. We present several examples of circuit design problems to demonstrate the power of DQAS. For instance, unitary operations are decomposed into quantum gates, noisy circuits are re-designed to improve accuracy, and circuit layouts for quantum approximation optimization algorithm are automatically discovered and upgraded for combinatorial optimization problems. These results not only manifest the vast potential of DQAS being an essential tool for the NISQ application developments, but also present an interesting research topic from the theoretical perspective as it draws inspirations from the newly emerging interdisciplinary paradigms of differentiable programming, probabilistic programming, and quantum programming.

1. Introduction

In the noisy intermediate-scale quantum technology (NISQ) era [1], the hybrid quantum–classical (HQC) computational scheme, combining quantum hardware evaluations with classical optimization outer loops, is widely expected to deliver the first instance of quantum advantages (for certain non-trivial applications) in the absence of fault-tolerant quantum error corrections. Several prototypical examples in this category include finding the ground state of complex quantum systems by variational quantum eigensolver (VQE) [2–4], exploring better approximation for NP hard combinatorial optimization problems by quantum approximation optimization algorithms (QAOA) [5–7], and solving some learning tasks in either the classical or quantum context by the quantum machine learning setup [8–12].

Under the typical setting in the HQC computational paradigm, the structure of variational ansatz is held fixed and only trainable parameters are optimized to satisfy an objective function. This lack of flexibility is rather undesirable as different families of parametrized circuits may differ substantially in their expressive power and entangling capability [13, 14]. Moreover, in the NISQ era, a thoughtful circuit design should minimize the consumption of quantum resources due to decoherence and limited connectivity among qubits in current quantum hardwares. For instance, the number of two-qubits gates (or the circuit depth) should be minimized to reduce noise-induced errors. Additional error mitigation strategy should be conducted without using extra qubits if possible. With these requirements in mind, the design of an effective circuit ansatz should take into account of the nature of the computational problems and the specifications of a quantum hardware as well. We term the automated design of parameterized circuits, in the aforementioned setting, as quantum *ansatz* search (QAS).

In a broader context, we denote QAS as quantum *architecture* search, which covers all scenarios of quantum circuit design and goes beyond the design of a variational ansatz for HQC algorithms. QAS can facilitate a broad range of tasks in quantum computations. Its applications include but not limited to decomposing arbitrary unitary [15] into given quantum gates, finding possible shortcuts for well-established quantum algorithms [16, 17], exploring optimal quantum control protocols [18–20], searching for powerful and resource-efficient variational ansatz [21], and designing end-to-end and task-specific circuits which also incorporate considerations on quantum error mitigation (QEM), native gate set, and topological connectivity of a specific quantum hardware [22, 23].

Neural architecture search (NAS) [24], devoted to the study and design of neural networks shares many similarities with designing parameterized quantum circuits. The common approaches for NAS include greedy algorithms [25], evolutionary or genetic algorithms [26–30], and reinforcement learning (RL) based methods [31–34]. It is interesting to witness that the progress in QAS follows closely the ideas presented in NAS. Recent works on quantum circuit structure or ansatz design also exploited greedy methods [35–37], evolutional or genetic methodologies [16, 17, 21–23, 38] and RL engine based approaches [19, 39] for tasks such as quantum control, QEM or circuit ansatz searching.

Recently, differentiable neural architecture search (DARTS) has been proposed [40] and further refined with many critical improvements and generalizations [41–46]. The key idea of a differentiable architecture search is to relax the discrete search space of neural architectures onto a continuous and differentiable domain, rendering much faster end-to-end NAS workflow than previous methods. Due to the close relation between NAS and QAS, it is natural to ask whether it is possible to devise a differentiable quantum architecture search (DQAS) incorporating DARTS-like ideas. Our answer is affirmative; as presented in this work, we constructed a general framework of DQAS that works very well as a universal and fully automated design tool for quantum circuits. As a general framework sitting at the intersection of newly emerging interdisciplinary paradigms of differentiable programming, probabilistic programming and quantum programming, DQAS is of both high theoretical and practical values across various fields in quantum computing and quantum information processing.

The organization of this work goes as follows. In background and related work section, we review backgrounds and relevant works on fields including NAS, QAS and QAOA. In methods section, we introduce the setup of the DQAS algorithm, where the overall workflow and the main components are both discussed. In applications section, we demonstrate various applications in quantum computing domain enabled by DQAS, including QEM and variational quantum algorithm design examples. We conclude with a brief discussion section. The appendix contains more details and further applications of DQAS [47].

2. Background and related work

Differentiable neural architecture search. NAS [48] is a burgeoning and active field in AutoML, and the ultimate goal of NAS is to automate the search for a top-performing neural network architectures for any given task. Popular approaches to implement NAS include reinforcement learning [31], in which an RNN controller chooses an action on building the network structure layerwise from a discrete set of options; and evolutionary or genetic algorithms [27, 28, 30], in which a population of network architectures is kept, evaluated, mutated for the fittest candidates. Such RL or evolutionary algorithms are rather resource intensive and time consuming, since the core task involves searching through an exponentially large space of discrete choices for different elementary network components.

Recently, differentiable architecture search [40] and its variants have been proposed and witnessed a surge in the number of related NAS studies [41–46, 49–57]. Under the DARTS framework, the network architecture space of discrete components is relaxed into a continuous domain that facilitates search by differentiation and gradient descent. The relaxed searching problem can be efficiently solved with noticeably reduced training time and hardware requirements.

In the original DARTS, the search space concerns with choices of distinct microstructures within one cell. Two types of cell are assumed for the networks: normal cell and reduction cell. The NAS proceeds by first determining the microstructures within these two types of cell, then a large network is built by stacking these two cell types up to a variable depth with arbitrary input and output size. Within each cell, two inputs, four intermediate nodes and one output (concatenation of four intermediate nodes) are presented as nodes in a directional acylic graph. For each edge between nodes, one needs to determine optimal connection layers, e.g. conv with certain kernel size, or max/average pooling with given window size, zero/identity connections and so on. To make such search process differentiable, we assume each edge is actually the weighted sum of all these primitive operations from the pool, i.e. $o(x) = \sum_i \mathrm{softmax}(\alpha_i)o_i(x)$ where o_i stands for ith type of layers primitives and α_i is the continuous weights which determines the structure of neural network as structural parameters. Therefore, we have two sets of continuous parameters:

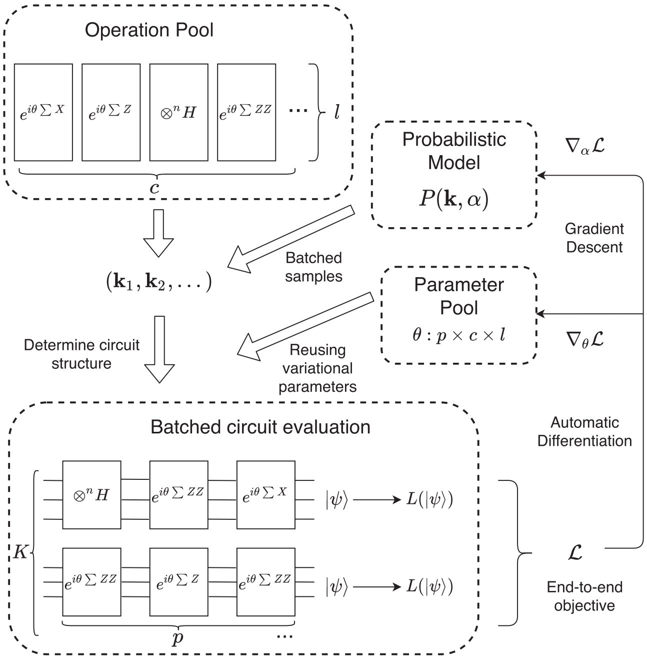

Operation Pool
$e^{i\theta\sum X}$
$e^{i\theta\sum Z}$
$\otimes^n H$
$e^{i\theta\sum ZZ}$
$\cdots$
l
c
Probabilistic Model
$P(\mathbf{k},\alpha)$
$\nabla_\alpha\mathcal{L}$
Gradient Descent
Batched samples
$(\mathbf{k}_1,\mathbf{k}_2,\ldots)$
Parameter Pool
$\theta : p\times c\times l$
$\nabla_\theta\mathcal{L}$
Determine circuit structure
Reusing variational parameters
Batched circuit evaluation
Automatic Differentiation
K
$|\psi\rangle \rightarrow L(|\psi\rangle)$
$\mathcal{L}$
End-to-end objective
p
$\cdots$

Figure 1. Schematic illustration of DQAS. We sample a batch of circuit configurations for each epoch from probabilistic model $P(\boldsymbol{k},\boldsymbol{\alpha})$. We then compose corresponding quantum circuits by filling in operations and parameters from two pools. We can evaluate quantum circuits and compute final objective $\mathcal{L}$ where $\boldsymbol{\alpha},\boldsymbol{\theta}$ can be adjusted accordingly with gradient based optimization method.

structure weights α which determines the optimal network architecture by pruning in evaluation stage, and conventional parameters in neural network ω. Via DARTS setup, neural architecture search turns into a bi-optimization problem where differentiation is carried out end-to-end.

DARTS requires the evaluation of the whole super network where each edge is composed of all types layers. This is memory intensive and limits its usage on large dataset or enriched cell structures. Therefore, there are works extending DARTS idea while enabling forward evaluation on sub network, usually using only one path [46, 57] or two [52]. Specifically, in [46], the authors viewed the super network as a probabilistic ensemble of subnetworks and thus variational structural parameters enter into NAS as probabilistic model parameters instead. So we can sample subnetworks from such probabilistic distribution and evaluate one subnetwork each time. This is feasible as probabilistic model parameters can also be updated from general theory for Monte Carlo expectations' gradient in a differentiable approach [58].

There are additional follow-up works that focus on improving drawbacks of DARTS with various training techniques. In general, these DARTS-related techniques are also illuminating and inspirational for further DQAS developments in our work.

Related works on QAS. Quantum architecture search, though no one brand it as this name before, is scattered in the literature with different contexts. These works are often specific to problem setup and denoted as quantum circuit structure learning [35], adaptive variational algorithm [36], ansatz architecture search [37], evolutional VQE [21], multipleobjective genetic VQE [22] or noise-aware circuit learning [23]. The tasks they focused are mainly in QAOA [37] or VQE [21, 22, 35, 36] settings. From higher theoretical perspective, some quantum control works can also be classified as QAS tasks, where optimal quantum control protocol is explored using possible machine learning tools [19, 39].

These QAS relevant works are closed related to NAS methodologies. And this relevance is as expected, since quantum circuit and neural network structure share a great proportion of similarities. The mainstream approach of QAS is evolution/genetic algorithms with different variants on mutation, crossover or tournament details [16, 17, 21–23, 38]. There are also works exploiting simple greedy/locality ideas [35–37] and reinforce learning ideas [19, 39] such as policy gradient.

All of the QAS works mentioned above are still searching quantum ansatz/architecture in discrete domain, which increases the difficulty on search and is in general time consuming. Due to the close relation between QAS and NAS together with the great success of differentiable NAS ideas in machine learning, we here introduce a framework of differentiable QAS that enable end-to-end automatic differentiable QAS (DQAS). This new approach unlocks more possibilities than previous works with less searching time and more versatile capabilities. It is designed with general QAS philosophy in mind, and DQAS framework is hence universal for all types of circuit searching tasks, instead of focusing only on one type of quantum computing tasks as previous work.

Brief review on QAOA. As introduced in [5], QAOA is designed to solve classical combinatorial optimization (CO) problems. These problems are often NP complete, such as MAX CUT or MIS in the graph theory. The basic idea is that we prepare a variational quantum circuit by alternately applying two distinct Hamiltonian evolution blocks. Namely, a standard QAOA anstaz reads

$$|\psi\rangle = \prod_{j=0}^{P} (\mathrm{e}^{\mathrm{i}H_{\mathrm{c}}\gamma_j}\,\mathrm{e}^{\mathrm{i}H_{\mathrm{b}}\beta_j})|\psi_0\rangle, \tag{1}$$

where $|\psi_0\rangle$ should be prepared in the space of feasible solutions (better as even superposition of all possible states, and in MAX CUT case $|\psi_0\rangle = \otimes H^n|0^n\rangle$, where n is the number of qubits and H is transversal Hadamard gates.)

In general, H_{c} is the objective Hamiltonian as $H_{\mathrm{c}}|\psi\rangle = f(\psi)|\psi\rangle$, where $f(\psi)$ is the CO objective. For MAX CUT problem on weighted graph with weight ω_{ij} on edge ij, $H_{\mathrm{c}} = -\sum \omega_{ij}Z_iZ_j$ up to some unimportant phase. (We use the notation $X/Y/Z_i$ for Pauli operators on ith qubit throughout this work) H_{b} is the mixer Hamiltonian to tunnel different feasible solutions, where $H_{\mathrm{b}} = \sum_{i=0}^{n} X_i$ is the most common one when there is no limitation on feasible Hilbert space.

The correctness of such ansatz is guaranteed when p approach infinity as it can be viewed as quantum annealing (QA), where we start from the ground state of Hamiltonian H_{b} as $|+^n\rangle$ and go through adiabatically to another Hamiltonian H_{c}, then it is expected that the final output state is the ground state of H_{c} which of course has the minimum energy/objective and thus solve the corresponding CO problems.

If we relax the strong restrictions from the QA limit and just take QAOA as some form of variational ansatz, then there are four Hamiltonians instead of two defining the ansatz.

- H_{g} the preparation Hamiltonian: we should prepare the initial states from zero product to the ground state of H_{g}. In original case, H_{g} is the same as H_{b}.
- H_{b} the mixer Hamiltonian: responsible to make feasible states transitions happen.
- H_{p} the phase/problem Hamiltonian: time evolution under the phase Hamiltonian and the mixer Hamiltonian alternately makes the bulk of the circuit, in original QAOA, H_{p} is the same as H_{c}.
- H_{c} the cost Hamiltonian: the Hamiltonian used in objectives and measurements where $\langle\psi|H_{\mathrm{c}}|\psi\rangle$ is optimized.

Moreover, such four Hamiltonian generalization of original QAOA can be further extended. For example, $H_{\mathrm{b}}, H_{\mathrm{p}}$ are not necessarily the same Hamiltonian for each layer of the circuit. Nonetheless, the essence of such ansatz is that: the number of variational parameters is of order the same as layer number P which is much less than other variational ansatz of the same depth such as typical hardware efficient VQE or quantum neural network design. This fact renders QAOA easier to train than VQE of the same depth and suffers less from barren plateaus [59]. And as QAOA ansatz has some reminiscent from QA, the final ansatz has better interpretation ability than typical random circuit ansatz. It is an interesting direction to automatically search for the four definition Hamiltonians or even more general layouts beyond vanilla QAOA, to see whether there are similar quantum architectures that can outperform vanilla QAOA in CO problems, this is where DQAS plays a role.

The physical intuition behind such QAOA type ansatz relaxation and searching originates from the close relation between QAOA and quantum adiabatic annealing. In particular, we draw inspirations from efforts to optimize annealing paths and boost performance for quantum annealers. We reckon at least two fronts to search for better ansatz for the HQC algorithm. The first case is to actually inspect the standard QAOA (which typically uses two alternating Hamiltonians to build the ansatz) and inquire if any ingredient may be improved. For instance, given the four Hamiltonians for the quantum-adiabatic inspired ansatz, one may search for a better initial-state-preparation Hamiltonian, or find better mixer Hamiltonians than the plain $\sum_i X_i$ for specific problems. Another inspiration derives from attempts to speed up quantum adiabatic annealing via ideas like catalyst Hamiltonians [60, 61], counter diabatic Hamiltonians [62, 63], and other ideas in shortcut to adiabaticity. These ultrafast annealing methods would entail design of complex annealing schedules that deviate from the simple linear schedule interpolating between an initial Hamiltonian and the target Hamiltonian. When these complex annealing paths are digitalized and projected onto the quantum gate model with variational approximations, they may just live in the form of XX Hamiltonians or local Y Hamiltonians. With these extra Hamiltonians, catalyst or counter diabatic, we anticipate better performances with shallower QAOA-like circuits layout may be achieved.

Algorithm 1. Differentiable quantum architecture search.

```
Require: $p$ as the number of components to build the circuit; operation pool with $c$ distinct unitaries; probabilistic model and its parameters $\boldsymbol{\alpha}$ with shape $p \times c$ initialized all to zero; reusing parameter pool $\boldsymbol{\theta}$ initialized with given initializer with shape $p \times c \times l$, where $l$ is the max number of parameters of each op in operation pool.
1: while not converged do
2:     Sample a batch of size $K$ circuits from model $P(k, \boldsymbol{\alpha})$.
3:     Compute the objective function for each circuit in the batch in the form of equations (3), (5) and (6) depending on different problem settings.
4:     Compute the gradient with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ according to equations (8) and (9), respectively.
5:     Update $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ using given gradient based optimizers and learning schedules.
6: end while
7: Get the circuit architecture $\boldsymbol{k}^*$ with the highest probability in $P(\boldsymbol{k}, \boldsymbol{\alpha})$; and fine tuning the circuit parameters as $\boldsymbol{\theta}^*$ associated with this circuit if necessary.
8: return final optimal circuit structure labeled by $\boldsymbol{k}^*$ and the associating weights $\boldsymbol{\theta}^*$.
```

3. Methods

Overview. The task of DQAS is to select several unitaries to compose the circuit that minimize the corresponding objective for a given task. The aim of DQAS is two-fold: one the one hand, DQAS determines a potentially optimal circuit layout, on the other hand, it also identifies suitable variational parameters for the circuit. To achieve the two goals at the same time, we regard DQAS as a bi-optimization problem, where both the parameters determining the quantum structure and trainable weights on the parameterized circuit are optimized via some gradient-based optimizers. To enable gradient descent search on the quantum structure, we relax the discrete structure parameters into continuous domain, where quantum architecture are viewed as the sample from some parameterized probabilistic model.

DQAS is presented as algorithm 1 with a visualized workflow in figure 1. We introduce the ingredients for DQAS and the general workflow below. (See appendix A for more details and the glossary of DQAS algorithm [47]).

Circuit encoding and operation pool. Any quantum circuit is composed of a sequence of unitaries with and without trainable parameters, i.e.

$$U = \prod_{i=0}^{p} U_i(\boldsymbol{\theta}_i), \tag{2}$$

where $\boldsymbol{\theta}_i$ can be of zero length corresponding to the case that U_i is a fixed unitary gate. Hence, we formulate the framework to cover circuit-design tasks beyond searching of variational ansatz.

In the most general term, these U_i can represent an one-qubit gate, a two-qubit gate or a higher-level block encoding, such as $\mathrm{e}^{\mathrm{i}H\theta}$ with a pre-defined Hermitian Hamiltonian H. This set of possible unitaries U_i constitutes the operation pool for DQAS, and the algorithm attempts to assemble a quantum circuit by stacking U_i together in order to optimize a task-dependent objective function.

We call the choice of primitive unitary gates in the operation pool along with the circuit layout of these gates a circuit encoding. In the operation pool, there are c different unitaries V_j, and each placeholder U_i should be assigned one of these V_j by DQAS. In this work, we refer to the placeholder U_i as the ith layer of the circuit U, no matter such placeholder actually stands for layers or other positional labels. The circuit design comes with replacement: one V_j from the operation pool can be used multiple times in building a single circuit U.

Objectives. To enable an end-to-end circuit design, a suitable objective should be specified. Such objectives are typically just sum of expectation values of some observables for HQC scenarios, such as combinatorial optimization problems or quantum simulations. Namely, the objective in these cases reads

$$L = \langle 0|U^\dagger H U|0\rangle, \tag{3}$$

where H is some Pauli strings such as $H = -\sum_{\langle ij\rangle} Z_i Z_j$ for MAX CUT problems and $|0\rangle$ represents the direct product state. This loss function L can be easily estimated by performing multiple shots of sample measurements in quantum hardwares.

However, the objectives can assume more general forms for a HQC algorithm. For instance, one may define more sophisticated objectives that not only depend on the mean value of measurements but also depend on distributions of different measurements. Examples include CVaR [64] and Gibbs objective [37], proposed to improve quality of solutions in QAOA. In general, DQAS-compatible objectives for HQC algorithms assume the following form,

$$L = \sum_i g_i(\langle 0|U^\dagger f_i(H_i) U|0\rangle), \tag{4}$$

where f_i and g_i are differentiable functions and H_i are Hermitian observables.

Extending the HQC algorithm to supervised machine learning setup that is commonly used in classification tasks, the objective function has to be further generalized to incorporate quantum-encoded data $|\psi_j\rangle$ with corresponding label y_j,

$$L = \sum_j \left(\sum_i g_i(\langle \psi_j | U^\dagger f_i(H_i) U | \psi_j \rangle) - y_j \right)^2. \tag{5}$$

Beyond ansatz searching for HQC algorithms, DQAS can be used to design circuits for state preparation or circuit compilations. In these scenarios, the objective is often taken as the fidelity between the proposed circuit design and a reference circuit, and the objective for pure states now reads

$$L = \sum_j \langle \phi_j | U | \psi_j \rangle, \tag{6}$$

where $|\phi_j\rangle = U_{\text{ref}}|\psi_j\rangle$ is the expected output of a reference circuit when $|\psi_j\rangle$ is the input state. For a state-preparation setup, the objective above is reduced to $L = \langle \phi | U | 0 \rangle$, where $|\phi\rangle$ is the target state. It is worth noting that the overlap objective can induce barren plateau issues and the local version of Hilbert–Schmidt test can be used as objectives to avoid barren plateaus [65, 66]. For a general task of unitary learning or compilation, the dimension of $|\psi_j\rangle$ can be as large as 2^n where n is the qubits number, such condition may be relaxed by sampling random inputs from Haar measure [67], which follows the philosophy of machine learning, especially stochastic batched gradient descent.

Sampling the structures. With circuit encoding and operation pool, the task of DQAS is reduced to assign p unitaries (selected from the operation pool) to the placeholder U_i in order to construct a circuit U that minimizes an objective $L(U)$. To facilitate the architecture search, it is tempting to relax the combinatorial problem into a continuous domain, amenable to optimization via gradient descent. We thus propose to embed the discrete structural choices into a continuously-parameterized probabilistic model. For instance, we consider a probabilistic model $P(\boldsymbol{k}, \boldsymbol{\alpha})$ where $\boldsymbol{k}$ is the discrete structural parameter determining the quantum circuits structure and hence $\boldsymbol{k}$ is often denoted as an intermediate representation (IR) for quantum circuit structure. For example, if IR $\boldsymbol{k} = [1, 3, 1]$ then it implies that the circuit structure $U(\boldsymbol{k}) = V_1 V_3 V_1$ where V_1 and V_3 refer to elements in the predefined operation pool introduced earlier. In the context of equation (2), $U_1 = V_1$, $U_2 = V_3$ and $U_3 = V_1$. $\boldsymbol{\alpha}$ is the continuous variable characterizing the distribution of the probabilistic model P. For naïve mean field probabilistic model, α_{ij} stands for the logarithmic probability to place V_j operator on the position of U_i placeholder. By such a design, we replace the intimidating task of searching for optimal structure in discrete IR space $\boldsymbol{k}$ with the easier task of optimizing continuous model parameters $\boldsymbol{\alpha}$.

In short, discrete random variables $\boldsymbol{k}$ are sampled from a probabilistic model characterized by parameters $\boldsymbol{\alpha}$. A particular $\boldsymbol{k}$ determines the structure of the circuit $U(\boldsymbol{k})$, and this circuit is used to evaluate the objectives $L(U)$. The final end-to-end objectives for DQAS reads

$$\mathcal{L} = \sum_{\boldsymbol{k} \sim P(\boldsymbol{k}, \boldsymbol{\alpha})} L(U(\boldsymbol{k}, \boldsymbol{\theta})). \tag{7}$$

And $\mathcal{L}$ depends indirectly on both variational circuit parameters $\boldsymbol{\theta}$ and probabilistic model parameters $\boldsymbol{\alpha}$, which can be both trained via gradient descent using automatic differentiation. The summation in (7) is over random circuit structures sampled from the distribution P.

Filling the circuit parameters. Since DQAS needs to sample multiple circuits U before deciding whether the current probabilistic model is ideal, we adopt the circuit parameter sharing mechanism for parametrized operators in the operation pool. We store a tensor of parameters $\boldsymbol{\theta}$ with size $p \times c \times l$, where p is the total number/layer of unitary placeholders to build the circuit, c is the size of the operation pool and l is the largest number of parameters for each unitaries in the operation pool, we denoted this tensor as a circuit parameter pool.

For example, if we place the jth operator V_j on the position of placeholder U_i as defined in equation (2), then we should fill such parameterized operator of l parameters with l values from parameter pool: $\theta[i, j, :]$. Therefore, every sampled parametrized V_j should be initialized with l parameters taken from the circuit parameter pool depending on the placeholder index i and its operation-pool index j. With this circuit parameter sharing mechanism, the variational parameters we need to maintain in architecture search is reduced from lc^p to lcp, i.e. an exponential reduction of trainable weights in total. This is the key to enabling a large scale quantum architecture search in terms of the operation pool size and the depth of the circuit. The number of possible quantum architectures is still exponential as c^p. However, this exponential scaling in terms of operation pool size is not a severe issue as: (1) the operation pool can be highly customizable and small enough by considering high-level encodings and (2) the exponential space can still be efficiently reached via Monte Carlo sampling from a informed probabilistic model. Therefore, the introduction of parameter sharing and architecture sampling render DQAS as a highly scalable approach for architecture search with moderate resources.

Quantum and Monte Carlo gradients. DQAS needs to optimize two sets of parameters, $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$, in order to identify a potentially ideal circuit for the task at hand. The gradients with respect to trainable circuit parameters $\boldsymbol{\theta}$ are easy to determine

$$\nabla_{\boldsymbol{\theta}}\mathcal{L} = \sum_{\boldsymbol{k}\sim P(\boldsymbol{k},\boldsymbol{\alpha})} \nabla_{\boldsymbol{\theta}} L(U(\boldsymbol{k},\boldsymbol{\theta})). \tag{8}$$

$\nabla_{\boldsymbol{\theta}} L(U)$ can be obtained with automatic differentiation in a classical simulation and from parameter shift [68] or other analytical gradient measurements [69] in quantum experiments.

As explained in algorithm 1, not all θ parameters would be present in a circuit which are sampled according to the probability $P(\boldsymbol{k},\boldsymbol{\alpha})$ at every iteration. For missing parameters in a particular circuit, the gradients are simply set to 0 as anticipated.

Calculations of gradients for $\boldsymbol{\alpha}$ should be treated more carefully, since these parameters are directly related to the outcomes of the Monte Carlo sampling from $P(\boldsymbol{k},\boldsymbol{\alpha})$. The calculation of gradient for the Monte Carlo expectations is an extensively studied problem [58] with two possible mainstream solutions: score function estimator [70] (also denoted as REINFORCE [71]) and pathwise estimator (also denoted as reparametrization trick [72]). In this work, we utilize the score function approach as it is more general and bears the potential to support calculations of higher order derivatives if desired [73, 74]. For unnormalized probabilistic model, the gradient with respect to $\boldsymbol{\alpha}$ is given by [47]

$$\nabla_{\alpha}\mathcal{L} = \sum_{\boldsymbol{k}\sim P} \nabla_{\alpha} \ln P(\boldsymbol{k},\boldsymbol{\alpha})\, L(U(\boldsymbol{k},\boldsymbol{\theta})) - \sum_{\boldsymbol{k}\sim P} L(U(\boldsymbol{k},\boldsymbol{\theta})) \sum_{\boldsymbol{k}\sim P} \nabla_{\alpha} \ln P(\boldsymbol{k},\boldsymbol{\alpha}). \tag{9}$$

For normalized probability distributions, $\langle \nabla_{\alpha} \ln P\rangle_P = 0$ and we may simply focus on the first term. Gradient of $\ln P$ can be easily evaluated via backward propagations on the given well-defined probabilistic model. By considering baseline trick to reduce the estimation variance, a batch size in the order of 10 is enough for a success DQAS training.

Probabilistic models. Throughout this work, we utilize the simplest probabilistic models: independent category probabilistic model, also known as naïve mean field model in energy model context. We stress that more complicated models such as the energy based models [9, 75, 76] and autoregressive models [77–80] may yield better performances under certain settings where explicit correlation between circuit layers is important. Such sophisticated probabilistic models can be easily incorporated into DQAS, and we leave this investigation as a future work.

The independent categorical probabilistic model we utilized is described as:

$$P(\boldsymbol{k},\boldsymbol{\alpha}) = \prod_{i=1}^{p} p(k_i, \boldsymbol{\alpha_i}), \tag{10}$$

where the probability p in each layer is given by a softmax

$$p(k_i = j, \boldsymbol{\alpha_i}) = \frac{\mathrm{e}^{\alpha_{ij}}}{\sum_k \mathrm{e}^{\alpha_{ik}}}, \tag{11}$$

where $k_i = j$ means that we pick $U_i = V_j$ from the operation pool, and the parameters $\boldsymbol{\alpha}$ are of the dimension $p \times c$. The gradient of such a probabilistic model can be determined analytically,

$$\nabla_{\alpha_{ij}} \ln P(k_i = m) = -P(k_i = m) + \delta_{jm}. \tag{12}$$

4. Applications

DQAS is a versatile tool for near-term quantum computations. In the following, we present several concrete examples to illustrate DQAS's potential to accelerate research and development of quantum algorithms and circuit compilations in the NISQ era [81]. Our implementation are based on quantum simulation backend of either Cirq [82]/TensorFlow Quantum [83] stack or TensorNetwork [84]/TensorCircuit [85] stack.

Firstly, it is natural to apply DQAS to quantum circuits design for state preparation as well as unitary decomposition. For example we can use DQAS to construct exact quantum circuit for GHZ state

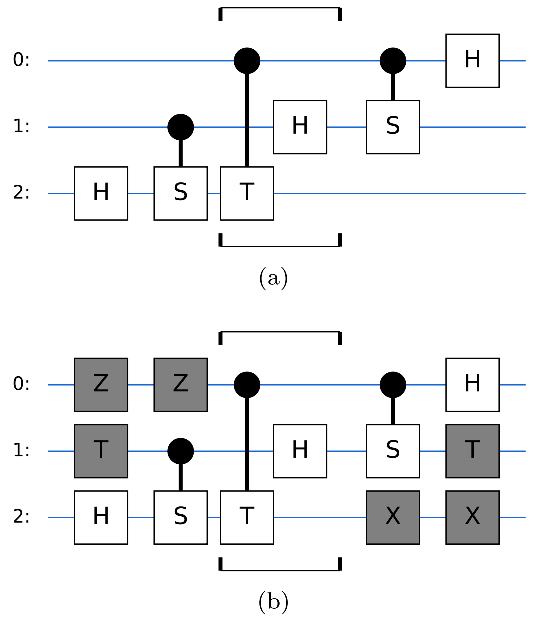


Figure 2. (a) The basic circuit for QFT on 3 qubits, T gate and H gate in the middle of the circuit can be easily arranged in the same vertical moment with no gap. And there are six gaps left (two on each qubit) in this setup. (b) The QEM circuit for QFT automatically found by DQAS. All slots are filled, DQAS is powerful enough to learn long range correlations so that it can fill the gaps on qubit 1 which are separately located. The fidelity between the two circuits on noisy hardwares and the ideal circuit are 0.33 and 0.6, respectively.

preparation or Bell circuit unitary decomposition [47]. We focus on QEM and HQC context in details below to demonstrate the power of DQAS for NISQ-relevant tasks.

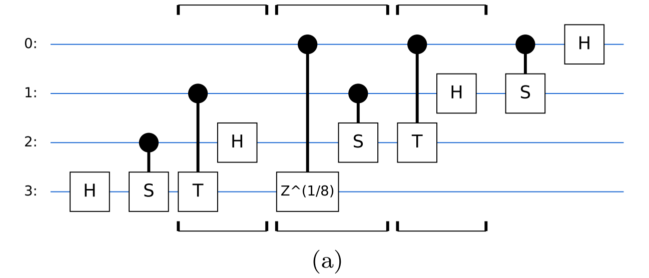
Quantum error mitigation on QFT circuit. Next, we demonstrate that DQAS can also be applied to design noise resilient circuits that mitigate quantum errors during a computation. The strategy we adopt in this work is to insert single qubit gates (usually Pauli gates) into the empty slots in a quantum circuit, where the given qubit are supposed to be found in idle/waiting status. Such gate-inserting technique can mitigate quantum errors since these extra unitaries (collectively act as an identity operation) can turn coherent errors into stochastic Pauli errors, which are easier to handle and effectively reduce the final infidelity. Similar QEM tricks are reported in related studies [86, 87].

The testbed is the standard circuit for quantum Fourier transformation (QFT), as shown in figure 2(a). We assume the following error model for an underlying quantum hardware. In between two quantum gates, there is a 2% chance of bit flip error incurred on a qubit. When a qubit is in an idle state (with much longer waiting time), there is a much higher chance of about 20% for bit flip errors. Although the error model is ad-hoc, it does not prevent us from demonstrating how DQAS can automatically design noise-resilient circuits.

Looking at figure 2(a), there are six empty slots in the standard QFT-3 circuit. Hence, we specify these slots as $p = 6$ placeholders for a search of noise-resilient circuits with DQAS. The search ends when DQAS fills each placeholder with a discrete single-qubit gate such that the fidelity of the circuit's output (with respect to the expected outcome) is maximized in the presence of noises.

If the operation pool is limited to Pauli gates and identity, $\{I, X, Y, Z\}$, then DQAS recommends a rather trivial circuit for error mitigation. In short, DQAS fills the pair gaps (of qubit 0 and qubit 2) with the same Pauli gate twice, which together yields an identity, in order to reduce the error in the gap. As for qubit 1, where a single gap occurs at the beginning and the end of the circuit as shown in figure 2(a), DQAS simply fills these gaps with nothing (identity placeholder). However, if we allow more variety of gates in the operation pool, such as S gate and T gate, then more interesting circuits can be found by DQAS. For instance, figure 2(b) is one such example. In this case, DQAS fills the two gaps of qubit 1 with a T gate each. This circuit cannot be found by the simpler strategy of inserting unitaries into consecutive gaps. Thus, DQAS provides a systematic and straightforward approach to identify this kind of long-range correlated gate assignments that should effectively reduce detrimental effects of noise.

We also carried out DQAS on QFT circuit for 4 qubits with $p = 12$ circuit gaps as shown in figure 3(a). DQAS automatically finds better QEM architecture which outperforms naïve gate inserting policies again. Figure 3(b) displays one such example. The interesting patterns of long-range correlated gate insertions are obvious for quibt 2. It is also clear that DQAS learns that more than two consecutive gates can combined



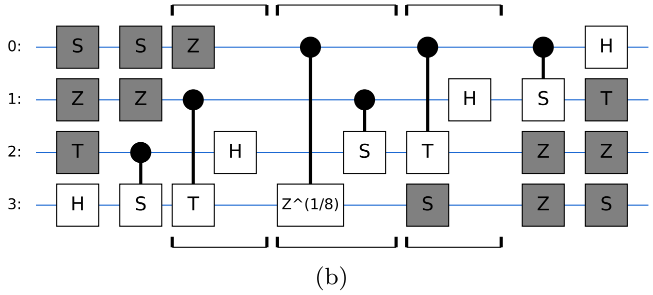


Figure 3. (a) The basic circuit for QFT on 4 qubits, some of the gates can be easily arranged in the same vertical moment with no gap. And there are 12 gaps left in this arrangement. (b) The QEM circuit for QFT automatically found by DQAS. The fidelity between the two circuits on noisy hardwares and the ideal circuit are 0.13 and 0.46, respectively.

collectively to render identity such as the three inserted gates for qubit 0. Further details on the search for optimal QEM architectures and comprehensive comparison on experiment values of final fidelities can be found in the appendix H [47].

In summary, DQAS not only learns about inserting pairs of gates as identity into the circuit to mitigate quantum error, but also picks up the technique of the long-range correlated gate assignment to further reduce the noise effects. This result is encouraging and shows how instrumental DQAS as a tool may be used for designing noise-resilient circuits with moderate consumption of computational resource. In this study, we only adapt the simple gate-insertion policy to design QEM within DQAS framework. We expect more sophisticated QEM methods may also be adapted to work along with DQAS to identify novel types of noise-resilient quantum circuits. This is a direction that we will actively explore in follow-up studies.

QAOA ansatz searching. QAOA introduces the adiabatic-process inspired ansatz that stacks alternating Hamiltonian evolution blocks as $\mathrm{e}^{-\mathrm{i}\theta H}$, where H could be different Hermitian Hamiltonians. QAOA can obtain better approximation ratio with increasing number of repetitive circuit blocks P as its infinite P limit is equivalent to quantum adiabatic evolution. To the end of employing DQAS to design parametrized quantum circuits within the HQC paradigm for algorithmic developments, we adopt a higher-level circuit encoding scheme as inspired by QAOA. More specifically, the operation pool consists of $\mathrm{e}^{-\mathrm{i}\theta H}$ blocks with different Hermitian Hamiltonians and also parameter free layers of traversal Hadamard gates $\otimes^n H$. In comparison to assembling a circuit by specifying individual quantum gates, this circuit encoding scheme allows a compact and efficient description of large-scale and deep circuits. For simplicity, we dub the circuit-encoding scheme above as the layer encoding.

For illustrations, we apply DQAS to design parametrized circuit for the MAXCUT problem in this subsection in QAOA-like fashion. Aiming to let DQAS find ansatz without imposing strong QAOA-type assumptions on the circuit architecture, we expand the operation pool with additional Hamiltonians of the form $\hat{H} = -\sum_{\langle ij\rangle} O_i O_j$ and $\hat{H} = \sum_i O_i$, where $O \in \{X, Y, Z\}$; and we refer to these operations as the xx-layer, rx-layer, rz-layer and so on. In addition, we also add the transversal Hadamard gates and denote it as the H-layer. All these primitive operations can be compiled into digital quantum gates exactly.

Next, let us elaborate on an interesting account that DQAS automatically re-discovers the standard QAOA circuit for the MAXCUT problem. To begin, we distinguish two settings: instance learning (for a single MAXCUT problem) and ensemble learning (for MAXCUT problems on ensemble of graphs). As noted in [88], the expected outputs by an ensemble of QAOA circuits (defined by graph instances from, say, Erdös–Rényi distributions or regular graph distributions) with fixed variational parameters θ are highly concentrated. The implication of such concentration is that the optimal parameters (for an arbitrary instance in the ensemble) can be quite close to being optimal for the entire ensemble of graph instances. This fact not only increases the stability of the learning process with an ensemble of data inputs, but also

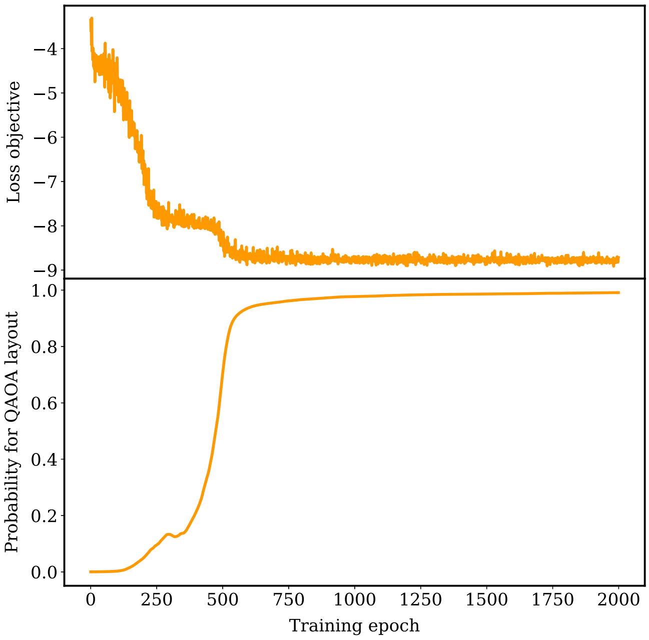


Figure 4. Metrics on DQAS training (depth $p = 5$) for MAX CUT problem of degree-3 regular graph ensemble with 8 nodes. The upper plot shows the expected energy/averaged cut value in the training process, the loss is approaching -8.8 which reflects the result from $P = 2$ QAOA with H, zz, rx, zz, rx layer arrangement. The lower plot indicates how the probability of such optimal layout is increased when the probabilistic model underlying is updated.

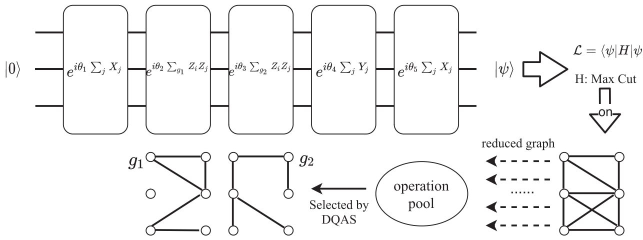

$|0\rangle$
$e^{i\theta_1 \sum_j X_j}$
$e^{i\theta_2 \sum_{g_1} Z_i Z_j}$
$e^{i\theta_3 \sum_{g_2} Z_i Z_j}$
$e^{i\theta_4 \sum_j Y_j}$
$e^{i\theta_5 \sum_j X_j}$
$|\psi\rangle$
$\mathcal{L} = \langle\psi|H|\psi\rangle$
H: Max Cut
on
reduced graph
......
g_1
g_2
Selected by DQAS
operation pool

Figure 5. Schematic workflow for reduced graph ansatz search on MAX CUT setup. In reduced ansatz searching, there are various reduced graph backend zz-layer in the operation pool. These reduced graph are sub graph instances from the problem graph instance. DQAS can not only find the optimal layout and optimal parameters $\boldsymbol{\theta}$, but also find the best reduced graph for these zz-layers.

makes QAOA more practical when the outer optimization loop can be done in this once-for-all fashion. In this work, we apply DQAS to both instance learning task and regular graph ensemble learning task [47].

For an ensemble learning on regular graph ensemble (node 8, degree 3), we let DQAS search for an optimal circuit design with $p = 5$. By using the aforementioned operation pool comprising the H-layer, rx/y/z-layer and zz-layer with the expected energy for the MAXCUT Hamiltonian as objective function, DQAS recommends the optimal circuit with the following layout: H, zz, rx, zz, rx layers, which coincides exactly with the original QAOA circuit. For metrics in the searching stage, see figure 4.

We also carried out DQAS on QAOA ansatz searching with multiple objective consideration on hardware details as well as double-layer block encoding for operations. For details, see the appendix I [47].

Reduced graph ansatz searching. To the end of designing circuits shallower than QAOA, another approach worth attempt is to re-define the primitive circuit layers in the operation pool. For instance, the zz-layer block is usually generated by the Ising Hamiltonian with the full connectivity of the MAXCUT problem. However, if the underlying graph of a zz-layer is only a subgraph then the number of gates would be reduced. Suppose we now replace the standard zz-layer (with full connectivity of the original problem) with a set of reduced zz-layers (each generated by a subgraph containing at most half of all edges in the original graph), then a circuit comprising 2 such reduced zz-layers is shallower than the standard $P = 1$ QAOA circuit. As summarized below, ansatz built from such reduced zz-layers is more resource efficient and outperforms the vanilla QAOA layout using the same number of quantum gates. Figure 5 summarizes the DQAS workflow in searching ansatz with reduced zz-layers.

To demonstrate the effectiveness of this strategy, we consider the circuit design under instance learning setup in which reduced zz-layers in the operation pool are induced by the graph connectivity of a particular instance. In this numerical experiment, we again set out to design a $p = 5$ circuit for $n = 8$ qubits. More specifically, we generate 10 subgraph with edge density lower than half of the base graph and substitute the base zz-layer with these 10 newly introduced reduced zz-layers in the operation pool. In such a setup, DQAS is responsible for finding (1) an optimal circuit layout of different types of layers, (2) best reduced graphs that give rise to the zz-layer in circuit, and (3) optimal parameters θ for rx/y/z-layer and zz-layer.

Here we give a concrete example. For an arbitrary graph instance drawn from the Erdös–Rényi distribution with a MAX CUT of 12, DQAS automatically design a circuit that exactly predicts the MAX CUT of 12. This $p = 5$ circuit is composed of following layers: rx-layer, zz-layer, zz-layer, ry-layer and rx-layer. Note the two zz-layers are induced by distinct sets of underlying subgraphs with only four edges each. As a comparison, the $P = 1$ vanilla QAOA gives expected MAX CUT of 10.39, while $P = 2$ vanilla QAOA predicts 11.18. In terms of overlap with exact MAX CUT configuration state, the reduced ansatz found by DQAS has nearly 100% success probability for one-shot measurement to get the MAX CUT value while $P = 2$ vanilla QAOA has 47% success probability to get the correct MAX CUT value. The reduced ansatz designed by DQAS consumes about the same amount of quantum resources as the $P = 1$ QAOA circuit yet even outperforms the vanilla $P = 2$ QAOA circuit. We stress that such an encouraging result is not a special case. By using the reduced ansatz layers, we can consistently find reduced ansatz that outperforms vanilla QAOA of the same depth for MAX CUT problems on a variety of unweighted and weighted graphs [47].

DQAS not only can learn QAOA from scratch, but also can easily find better alternatives with shorter circuit depth with an operation pool using slightly tweaked Hamiltonian evolution blocks as primitive circuit layers. This last achievement is of paramount importance in the NISQ era where circuit depth is a key limitation.

5. Discussions

DQAS is a versatile and useful tool in the NISQ era. Not only can DQAS handle the design of a quantum circuit, but it can also be seamlessly tailored for a specific quantum hardware with customized noise model and native gate set in order to get best results for error mitigation. We have demonstrated the potential of DQAS with the following examples: circuit design for state preparing and unitary decomposition (compilation), and noiseless and noisy circuit design for the hybrid quantum classical computations. In particular, we also introduce the reduced ansatz design that proposes shallower circuits that outperforms the conventional QAOA circuits that are inherently more resource intensive ansatz.

In conclusion, we re-formulate the design of quantum circuits and HQC algorithm as an automated DQAS. Inspired by DARTS-like setup in NAS, DQAS works in a differentiable search space for quantum circuits. By tweaking multiple ingredients in DQAS, the framework is highly flexible and versatile. Not only can it be used to design optimal quantum circuits under different scenarios but it also does the job in a highly customized fashion that takes into account of native gate sets, hardware connectivity, and error models for specific quantum hardwares. The theoretical framework itself offers a fertile ground for further study as it draws advanced concepts and techniques from the newly emerged interface of differential, probabilistic, and quantum programming paradigms.

Note added

After this work was posted on arXiv, a relevant paper [89] was also posted. This paper also proposed the idea of using quantum architecture search as a promising strategy for designing hardware-specific and noise-resilient quantum circuits. Conceptually, this work shares some similarities with our work. The approach utilized in their work is of random search and evolutionary nature, where the circuit sampling process stays evenly distributed (i.e. a fixed probabilistic model in our context) while our DARTS-inspired workflow iteratively updates both circuit parameters and the circuit-structure probabilistic model. Together, these two works validate the benefits of using QAS framework to optimize quantum circuits and should help substantially in establishing quantum advantage in the NISQ era.

Acknowledgments

SXZ and HY are supported in part by the NSFC under Grant No. 11825404. HY is also supported in part by the MOSTC under Grant Nos. 2016YFA0301001 and 2018YFA0305604, the Strategic Priority Research Program of Chinese Academy of Sciences under Grant No. XDB28000000, Beijing Municipal Science and Technology Commission under Grant No. Z181100004218001, and Beijing Natural Science Foundation under Grant No. Z180010.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

ORCID iDs

Chang-Yu Hsieh https://orcid.org/0000-0002-6242-4218

References

[1] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[2] Peruzzo A, McClean J, Shadbolt P, Yung M-H, Zhou X-Q, Love P J, Aspuru-Guzik A and O'Brien J L 2014 A variational eigenvalue solver on a photonic quantum processor *Nat. Commun.* **5** 4213
[3] McClean J R, Romero J, Babbush R and Aspuru-Guzik A 2016 The theory of variational hybrid quantum–classical algorithms *New J. Phys.* **18** 023023
[4] McArdle S, Endo S, Aspuru-Guzik A, Benjamin S C and Yuan X 2020 Quantum computational chemistry *Rev. Mod. Phys.* **92** 015003
[5] Farhi E, Goldstone J and Gutmann S 2014 A quantum approximate optimization algorithm (arXiv:1411.4028)
[6] Hadfield S, Wang Z, O'Gorman B, Rieffel E, Venturelli D and Biswas R 2019 From the quantum approximate optimization algorithm to a quantum alternating operator ansatz *Algorithms* **12** 34
[7] Zhou L, Wang S-T, Choi S, Pichler H and Lukin M D 2020 Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices *Phys. Rev.* X **10** 021067
[8] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)
[9] Verdon G, Marks J, Nanda S, Leichenauer S and Hidary J 2019 Quantum Hamiltonian-based models and the variational quantum thermalizer algorithm (arXiv:1910.02071)
[10] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001
[11] Benedetti M, Garcia-pintos D, Perdomo O, Leyton-Ortega V, Nam Y and Perdomo-Ortiz A 2019 A generative modeling approach for benchmarking and training shallow quantum circuits *npj Quantum Inf.* **5** 45
[12] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273
[13] Akshay V, Philathong H, Morales M E S and Biamonte J D 2020 Reachability deficits in quantum approximate optimization *Phys. Rev. Lett.* **124** 090504
[14] Farhi E, Gamarnik D and Gutmann S 2020 The quantum approximate optimization algorithm needs to see the whole graph: a typical case (arXiv:2004.09002)
[15] Kiani B T, Lloyd S and Maity R 2020 Learning unitaries by gradient descent (arXiv:2001.11897)
[16] Li R, Alvarez-Rodriguez U, Lamata L and Solano E 2017 Approximate quantum adders with genetic algorithms: an IBM quantum experience *Quantum Meas. Quantum Metrol.* **4** 1
[17] Cincio L, Subaşi Y, Sornborger A T, Coles P J, Sotnikov O M, Cincio L, Suba Y, Sornborger A T and Coles P J 2018 Learning the quantum algorithm for state overlap *New J. Phys.* **20** 113022
[18] Yang Z-C, Rahmani A, Shabani A, Neven H and Chamon C 2017 Optimizing variational quantum algorithms using Pontryagin's minimum principle *Phys. Rev.* X **7** 021027
[19] Fösel T, Tighineanu P, Weiss T and Marquardt F 2018 Reinforcement learning with neural networks for quantum feedback *Phys. Rev.* X **8** 31084
[20] Lin C, Wang Y, Kolesov G and Kalabić U 2019 Application of Pontryagin's minimum principle to Grover's quantum search problem *Phys. Rev.* A **100** 022327
[21] Rattew A G, Hu S, Pistoia M, Chen R and Wood S 2019 A domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver (arXiv:1910.09694)
[22] Chivilikhin D, Samarin A, Ulyantsev V, Iorsh I, Oganov A R and Kyriienko O 2020 MoG-VQE: multiobjective genetic variational quantum eigensolver (arXiv:2007.04424)
[23] Cincio L, Rudinger K, Sarovar M and Coles P J 2021 Machine learning of noise-resilient quantum circuits *PRX Quantum* **2** 010324
[24] Yao Q, Wang M, Chen Y, Dai W, Li Y-F, Tu W-W, Yang Q and Yu Y 2018 Taking human out of learning applications: a survey on automated machine learning (arXiv:1810.13306)
[25] Huang S, Li X, Cheng Z-Q, Zhang Z and Hauptmann A 2018 GNAS *2018 ACM Multimedia Conf. on Multimedia Conf.—MM '18* (New York: ACM) pp 2049–57
[26] Stanley K O, Clune J, Lehman J and Miikkulainen R 2019 Designing neural networks through neuroevolution *Nat. Mach. Intell.* **1** 24
[27] Real E, Moore S, Selle A, Saxena S, Suematsu Y L, Tan J, Le Q V and Kurakin A 2017 Large-scale evolution of image classifiers *34th Int. Conf. on Machine Learning* pp 2902–11
[28] Xie L and Yuille A 2017 Genetic CNN *Proc. IEEE Int. Conf. on Computer Vision* pp 1388–97

[29] Liu H, Simonyan K, Vinyals O, Fernando C and Kavukcuoglu K 2018 Hierarchical representations for efficient architecture search *6th Int. Conf. on Learning Representations, ICLR 2018*

[30] Real E, Aggarwal A, Huang Y and Le Q V 2019 Regularized evolution for image classifier architecture search *Proc. AAAI Conf. on Artificial Intelligence* vol 33 pp 4780–9

[31] Zoph B and Le Q V 2017 Neural architecture search with reinforcement learning *5th Int. Conf. on Learning Representations, ICLR 2017—Conf. Track Proc.*

[32] Baker B, Gupta O, Naik N and Raskar R 2017 Designing neural network architectures using reinforcement learning *5th Int. Conf. on Learning Representations, ICLR 2017—Conf. Track Proc.*

[33] Cai H, Chen T, Zhang W, Yu Y and Wang J 2018 Efficient architecture search by network transformation *32nd AAAI Conf. on Artificial Intelligence* pp 2787–94

[34] Zoph B, Vasudevan V, Shlens J and Le Q V 2018 Learning transferable architectures for scalable image recognition *CVPR* pp 8697–710

[35] Ostaszewski M, Grant E and Benedetti M 2021 Structure optimization for parameterized quantum circuits *Quantum* **5** 391

[36] Grimsley H R, Economou S E, Barnes E and Mayhall N J 2019 An adaptive variational algorithm for exact molecular simulations on a quantum computer *Nat. Commun.* **10** 3007

[37] Li L, Fan M, Coram M, Riley P and Leichenauer S 2020 Quantum optimization with a novel Gibbs objective function and ansatz architecture search *Phys. Rev. Res.* **2** 023074

[38] Las Heras U, Alvarez-Rodriguez U, Solano E and Sanz M 2016 Genetic algorithms for digital quantum simulations *Phys. Rev. Lett.* **116** 230504

[39] Niu M Y, Boixo S, Smelyanskiy V N and Neven H 2019 Universal quantum control through deep reinforcement learning *npj Quantum Inf.* **5** 33

[40] Liu H, Simonyan K and Yang Y 2019 DARTS: differentiable architecture search *7th Int. Conf. on Learning Representations, ICLR 2019*

[41] Xie S, Zheng H, Liu C and Lin L 2019 SNAS: stochastic neural architecture search *7th Int. Conf. on Learning Representations, ICLR 2019*

[42] Liang H, Zhang S, Sun J, He X, Huang W, Zhuang K and Li Z 2019 DARTS+: improved differentiable architecture search with early stopping (arXiv:1909.06035)

[43] Zela A, Elsken T, Saikia T, Marrakchi Y, Brox T and Hutter F 2019 Understanding and robustifying differentiable architecture search *ICLR*

[44] Hundt A, Jain V and Hager G D 2019 sharpDARTS: faster and more accurate differentiable architecture search (arXiv:1903.09900)

[45] Chen X, Xie L, Wu J and Tian Q 2019 Progressive differentiable architecture search: bridging the depth gap between search and evaluation *Proc. IEEE Int. Conf. on Computer Vision* pp 1294–303

[46] Casale F P, Gordon J and Fusi N 2019 Probabilistic neural architecture search (arXiv:1902.05116)

[47] See supplemental materials (https://stacks.iop.org/QST/7/045023/mmedia) for more information on: (A) glossary for ingredients of DQAS. (B) The connection between DARTS and DQAS. (C) Gradients derivation from Monte Carlo expectation in DQAS. (D) General hyperparameters for DQAS training. (E) Training techniques that stabilize and improve DQAS. (F) DQAS applications on state preparation and unitary learning. (G) Relevant hyperparameter and ingredient settings on experiments in this work. (H) More results and comparisons on QEM for QFT circuit. (I) More DQAS results for QAOA ansatz search including instance learning, block encoding, etc

[48] Ren P, Xiao Y, Chang X, Huang P-Y, Li Z, Chen X and Wang X 2020 A comprehensive survey of neural architecture search: challenges and solutions (arXiv:2006.02903)

[49] Dong X and Yang Y 2019 Searching for a robust neural architecture in four GPU hours *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition* pp 1761–70

[50] Yao Q, Xu J, Tu W-W and Zhu Z 2019 Efficient neural architecture search via proximal iterations (arXiv:1905.13577)

[51] Noy A, Nayman N, Ridnik T, Zamir N, Doveh S, Friedman I, Giryes R and Zelnik-Manor L 2019 ASAP: architecture search, anneal and prune (arXiv:1904.04123)

[52] Cai H, Zhu L and Han S 2019 Proxyless NAS: direct neural architecture search on target task and hardware *ICLR 2019*

[53] Li G, Qian G, Delgadillo I C, Müller M, Thabet A and Ghanem B 2019 SGAS: sequential greedy architecture search (arXiv:1912.00195)

[54] Xu Y, Xie L, Zhang X, Chen X, Qi G-J, Tian Q and Xiong H 2019 PC-DARTS: partial channel connections for memory-efficient architecture search *ICLR*

[55] Chang J 2019 DATA: differentiable ArchiTecture approximation *33rd Conf. on Neural Information Processing Systems (NeurIPS)*

[56] Chen X, Hsieh C-J, Regularization P-b and Hsieh X C C-j 2020 Stabilizing differentiable architecture search via perturbation-based regularization *Proc. 37th Int. Conf. on Machine Learning PMLR* (Vienna, Austria) vol 119

[57] Hu S, Xie S, Zheng H, Liu C, Shi J, Liu X and Lin D 2020 DSNAS: direct neural architecture search without parameter retraining (arXiv:2002.09128)

[58] Mohamed S, Rosca M, Figurnov M and Mnih A 2020 Monte Carlo gradient estimation in machine learning *J. Mach. Learn. Res.* **21** 1

[59] McClean J R, Boixo S, Smelyanskiy V N, Babbush R and Neven H 2018 Barren plateaus in quantum neural network training landscapes *Nat. Commun.* **9** 4812

[60] Crosson E, Farhi E, Lin C Y-Y, Lin H-H and Shor P 2014 Different strategies for optimization using the quantum adiabatic algorithm (arXiv:1401.7320)

[61] Albash T 2019 Role of nonstoquastic catalysts in quantum adiabatic optimization *Phys. Rev.* A **99** 042334

[62] Sels D and Polkovnikov A 2017 Minimizing irreversible losses in quantum systems by local counterdiabatic driving *Proc. Natl Acad. Sci. USA* **114** E3909

[63] Hartmann A and Lechner W 2019 Rapid counter-diabatic sweeps in lattice gauge adiabatic quantum computing *New J. Phys.* **21** 043025

[64] Barkoutsos P K, Nannicini G, Robert A, Tavernelli I and Woerner S 2020 Improving variational quantum optimization using CVaR *Quantum* **4** 256

[65] Khatri S, LaRose R, Poremba A, Cincio L, Sornborger A T and Coles P J 2019 Quantum-assisted quantum compiling *Quantum* **3** 140

[66] Sharma K, Khatri S, Cerezo M and Coles P J 2020 Noise resilience of variational quantum compiling *New J. Phys.* **22** 043006

[67] Nakata Y, Hirche C, Morgan C and Winter A 2017 Unitary two-designs from random X- and Z-diagonal unitaries *J. Math. Phys.* **58** 052203

[68] Crooks G E 2019 Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition (arXiv:1905.13311)

[69] Harrow A W and Napp J C 2021 Low-depth gradient measurements can improve convergence in variational hybrid quantum–classical algorithms *Phys. Rev. Lett.* **126** 140502

[70] Kleijnen J P C and Rubinstein R Y 1996 Optimization and sensitivity analysis of computer simulation models by the score function method *Eur. J. Oper. Res.* **88** 413

[71] Williams R J 1992 Simple statistical gradient-following algorithms for connectionist reinforcement learning *Mach. Learn.* **8** 229

[72] Kingma D P and Welling M 2014 Auto-encoding variational bayes *Proc. 2nd Int. Conf. on Learning Representations*

[73] Foerster J, Farquhar G, Rocktäschel T, Whiteson S, Al-Shedivat M and Xing E P 2018 DICE: the infinitely differentiable Monte Carlo estimator *Proc. 35th Int. Conf. on Machine Learning*

[74] Zhang S-X, Wan Z-Q and Yao H 2019 Automatic differentiable Monte Carlo: theory and application (arXiv:1911.09117)

[75] Hinton G E 2012 A practical guide to training restricted Boltzmann machines *Neural Networks: Tricks of the Trade* (Berlin: Springer) p 599

[76] Carleo G and Troyer M 2017 Solving the quantum many-body problem with artificial neural networks *Science* **355** 602

[77] Germain M, Gregor K, Murray I and Larochelle H 2015 MADE: masked autoencoder for distribution estimation *32nd Int. Conf. on Machine Learning, ICML 2015* pp 881–9

[78] Wu D, Wang L and Zhang P 2019 Solving statistical mechanics using variational autoregressive networks *Phys. Rev. Lett.* **122** 080602

[79] Sharir O, Levine Y, Wies N, Carleo G and Shashua A 2020 Deep autoregressive models for the efficient variational simulation of many-body quantum systems *Phys. Rev. Lett.* **124** 020503

[80] Liu J-G, Mao L, Zhang P and Wang L 2021 Solving quantum statistical mechanics with variational autoregressive networks and quantum circuits *Mach. Learn.: Sci. Technol.* **2** 025011

[81] Code implementation of DQAS and its applications can be found at https://github.com/refraction-ray/tensorcircuit/tree/master/tensorcircuit/applications

[82] See https://github.com/quantumlib/Cirq

[83] See https://github.com/tensorflow/quantum

[84] See https://github.com/google/tensornetwork

[85] See https://github.com/refraction-ray/tensorcircuit

[86] Wallman J J and Emerson J 2016 Noise tailoring for scalable quantum computation via randomized compiling *Phys. Rev.* A **94** 052325

[87] Zlokapa A and Gheorghiu A 2020 A deep learning model for noise prediction on near-term quantum devices (arXiv:2005.10811)

[88] Brandao F G S L, Broughton M, Farhi E, Gutmann S and Neven H 2018 For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances (arXiv:1812.04170)

[89] Du Y, Huang T, You S, Hsieh M-H and Tao D 2020 Quantum circuit architecture search: error mitigation and trainability enhancement for variational quantum solvers (arXiv:2010.10217)