MACHINE LEARNING
Science and Technology

Neural predictor based quantum architecture search

MACHINE
LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
6 July 2021

REVISED
7 August 2021

ACCEPTED FOR PUBLICATION
21 September 2021

PUBLISHED
8 October 2021

PAPER

Neural predictor based quantum architecture search

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

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

E-mail: kimhsieh@tencent.com and yaohong@tsinghua.edu.cn

Keywords: neural network, quantum circuit design, quantum algorithm

Supplementary material for this article is available online

Abstract

Variational quantum algorithms (VQAs) are widely speculated to deliver quantum advantages for practical problems under the quantum–classical hybrid computational paradigm in the near term. Both theoretical and practical developments of VQAs share many similarities with those of deep learning. For instance, a key component of VQAs is the design of task-dependent parameterized quantum circuits (PQCs) as in the case of designing a good neural architecture in deep learning. Partly inspired by the recent success of AutoML and neural architecture search (NAS), quantum architecture search (QAS) is a collection of methods devised to engineer an optimal task-specific PQC. It has been proven that QAS-designed VQAs can outperform expert-crafted VQAs in various scenarios. In this work, we propose to use a neural network based predictor as the evaluation policy for QAS. We demonstrate a neural predictor guided QAS can discover powerful quantum circuit ansatz, yielding state-of-the-art results for various examples from quantum simulation and quantum machine learning. Notably, neural predictor guided QAS provides a better solution than that by the random-search baseline while using an order of magnitude less of circuit evaluations. Moreover, the predictor for QAS as well as the optimal ansatz found by QAS can both be transferred and generalized to address similar problems.

1. Introduction

Variational quantum algorithms [1, 2] (with a relatively low consumption on quantum resources) are readily accessible in the noisy intermediate-scale quantum (NISQ) era [3] and show some promising signs for practical quantum advantage in the near future. Some prominent examples of VQA include variational quantum eigensolvers (VQE) [4–9], quantum approximation optimization algorithms (QAOA) [10–14], and variational quantum machine learning (QML) [15–20, 20–27]. Such parameterized quantum circuit (PQC) based variational algorithms have been exploited to solve many tasks that are considered hard for classical computers, such as integer factoring, combinatorial optimization, quantum Hamiltonian ground state approximation, and quantum dynamics simulation.

The conventional VQA requires a fixed quantum structure ansatz where the trainable parameters are iteratively adjusted via a classical optimizer in order to minimize an objective function. Some famous PQCs include QAOA ansatz for combinatorial problems [10], hardware efficient ansatz [28] and unitary coupled clusters ansatz [4, 5, 29] for VQE. There are also extensive studies on the expressive power and trainability of these ansatzes [30–34]. However, it still remains open on how to discover specifically tailored quantum ansatzes for different tasks. Partly inspired by neural architecture search from the AutoML community [35–38], we introduced the concept of quantum architecture search (QAS), which refers to a collection of effective methods that systematically search for an optimal quantum circuit ansatz for a given problem in reference [39]. Due to the apparent analogy between variational quantum circuits and neural networks, approaches developed in NAS have been adapted to QAS. Representative works have utilized ideas including evolutionary or genetic algorithms [40–46], reinforcement learning (RL) approaches [47, 48], greedy algorithms [9, 49–51], and differentiable architecture search (DARTS) [39].

QAS requires exploring and evaluating many quantum ansatzes during the training, and this computational bottleneck is intrinsic to all design-by-search methodologies including NAS in deep learning. In order to enhance search efficiency, there are two mainstream evaluation policies widely adopted in NAS. The first one is weight sharing, where trainable parameters are shared across different ansatzes instead of standalone training for each ansatz. Such weight sharing policy is utilized in one-shot search [52–54] and DARTS [55] in NAS, and the same idea has been exploited in the corresponding QAS frameworks, quantum circuit architecture search [56] and differentiable QAS [39], respectively. The second type of evaluation policy is to evaluate the fitness of an architecture by a meta machine learning (ML) model. Such predictor-based methods constitute another well-established and actively researched subfield in NAS [57–69]; yet, to the best of our knowledge, a predictor-based QAS framework has not been explored so far. In this work, we introduce the first predictor-based QAS. We train a neural predictor to directly gauge the performance of candidates of quantum circuits using only the structure of quantum ansatz. This predictor is then integrated into a QAS workflow (to be elucidated) and can substantially accelerate the search process.

The main contributions of the present work are summarized below.

(a) We introduce meta neural network predictor to a QAS framework, and achieve a dramatic boost on search efficiency compared to a random search baseline.
(b) We demonstrate that the proposed neural predictor based QAS performs competitively in some VQE and QML tasks as it efficiently discovers state-of-the-art quantum architectures.
(c) We also show that the optimal circuit structure suggested by QAS can be readily adapted to problems of different sizes. We develop a sophisticated transfer protocol incorporating beam search and evolutionary techniques to achieve this.

2. Related work

2.1. Predictor based NAS

NAS [35–38] is a recently emerging and rapidly developing field. It aims at automatically searching for optimal neural networks for some given tasks without relying on handcrafted design and expert guidance. An effective NAS workflow is composed of several ingredients, in which the sampling strategy and evaluation protocol are among the most important. In short, sampling strategy refers to a customized recommendation of candidate neural networks, whose fitness for a specific task should be evaluated by training the network and checking accuracy on some validation or test sets. Since the search space is exponentially large, we need an efficient method to traverse and sample candidate architectures. The sampling strategies that have been previously attempted include random search [70], local and greedy search [71–73], evolutionary and genetic type search [74–78], RL based search [79–82], Bayesian optimization search [59, 61, 62] and so on. While sampling is essential for NAS, when it comes to the computational cost, the bottleneck lies in the evaluation part as it is time consuming to train individual networks (from scratch) on a large dataset. Indeed, early NAS works [79] using such plain evaluation methods with RL or evolutionary engines often took thousands of GPU hours in the training process.

There are two approaches to lower the cost of individual evaluations. The first one is weight sharing. In this setup, all candidate networks are organized within a super network, which can be trained either by training the entire network or by training sampled subnetworks in each epoch. After training the super network, one can evaluate each candidate subnetwork by regarding it as a child of the super network, and the weights of such candidates are inherited from the super network directly without further tuning. Therefore, the evaluation on the candidate network is as simple as a forward pass for inference. Such parameter sharing setup is inspired by one-shot NAS [52–54] and has also become popular for DARTS [55, 83–85]. Parameter sharing is not perfect, though. As there is actually no theoretical guarantee on the accuracy correlation between subnetworks with inherited weights and optimal weights from individual training.

There is a second strategy to improve the evaluation methodology which our work focuses on. Instead of training a candidate network and evaluating its performance on some validation dataset, one directly builds ML models to predict the network performance based on the network structure alone, i.e. without specifying trainable weights. If the prediction accuracy manifests a non-trivial correlation with the ground truth, then such predictor based evaluation method may greatly ameliorate NAS efficiency. We call works along this line predictor based NAS, in which the so-called 'predictor' is a regression model [57–69]. Some works take a further step by constructing variational autoencoder (VAE) [86] for neural architectures and trains a predictor with input from the latent space of VAE [87–93]. Such predictor can also be transferred to infer other metrics of the network such as latency or FLOPs [59, 68]. It is worth noting that predictor based NAS can be combined with weight sharing tricks, where the predictor is trained with fitness labels obtained from one-shot setup [62, 65, 87, 88].

2.2. Variational quantum algorithms

The paradigm of variational quantum algorithms [1, 2] is widely speculated to deliver practical quantum advantages with NISQ devices [3] in the near term. The hybrid quantum–classical implementation of VQAs bears many similarities to deep learning in classical computations. Various VQAs have been designed for problems in a vast array of fields. Some typical examples include determining the ground and low excited states of a quantum Hamiltonian [4, 6–8, 94–97], simulating quantum dynamics [98–107], preparing Gibbs thermal states [108–110], supervised and unsupervised ML [15–20, 20–27], compiling quantum circuits [111–113], solving classical combinatorial optimizations [10–14], factoring integers [114–116], and solving linear or differential equations [117–121], etc. A VQA requires a fixed PQC ansatz at the beginning stage. Such fixed ansatz is usually designed to accommodate limitations of current hardware [28] or are inspired by well-established theoretical techniques, such as quantum annealing, unitary coupled-cluster, etc [4, 5, 10, 29, 122].

Despite a wide application range and initial successes, pushing for quantum advantages with VQA faces many challenges such as noise-induced decoherence, barren plateaus [123, 124] that derail the training of parameters, and reachability deficits with certain fixed ansatz [30–32, 34]. Although there are various proposals [98, 125–128] on alleviating these issues, it is extremely difficult to fully resolve them as long as the circuit ansatz is fixed at the beginning.

2.3. Quantum architecture search

As discussed in the last section, a parametrized quantum circuit is required as an ansatz for VQA. A badly designed ansatz could possess limited expressive power or entangling capacity, leaving the global minimum for an optimization problem out of reach. Furthermore, such ansatz may be more susceptible to noises [129], waste quantum resource or lead to barren plateaus that frustrate the optimization procedure [123, 124].

Therefore, a systematic approach to search for optimal circuit ansatz is desired, which has been studied in the name of 'quantum architecture search' (QAS) [39]. The aim of the QAS is to recommend tailored quantum circuits for given problems such that it not only minimizes a loss function, but also satisfies a few other constraints imposed by the hardware connectivity among qubits, native quantum gate set, quantum noise model, training loss landscape and other practical issues.

Previous QAS works have heavily borrowed ideas from NAS. More specifically, greedy methods [9, 49–51], evolutionary or genetic methodologies [40–46], RL engine based approaches [47, 48], Bayesian optimization [130], one-shot search [56], and gradient based methods [39] have all been adopted to discover better circuit ansatz for VQA. However, as far as we know, predictor based evaluation strategy has not been applied toward quantum circuit design. Since predictor based NAS has been empirically demonstrated to be highly efficient, one may anticipate predictor based QAS to hold a similar performance boost for VQA in the NISQ era.

3. Methods

In this section, we describe the essential technical ingredients of our proposed neural predictor based QAS. The predictor based QAS workflow we adopted in this work includes two phases: at phase I, we firstly generate a circuit dataset by standalone training and then build the neural predictor by training with this dataset; at phase II, we utilize the trained predictor to identify high-performance circuit candidates in the search space. The workflow is sketched as in figure 1.

Section 3 is organized as follows. In subsection 3.1, we discuss the search space of our proposed QAS and the corresponding list representation for quantum circuits, which defines the search space and is used to sample search candidates. In subsection 3.2, we present the other representation of quantum circuits, i.e. image representation, which is used as input for the neural predictor. In subsections 3.3 and 3.4, we show how to construct and train the neural predictor, respectively. Equipped with all the above ingredients, we give the overall workflow for predictor based QAS in subsection 3.5. Finally, we address how to transfer optimal ansatz to problems of different sizes based on beam search in subsection 3.6.

To facilitate the discussion, we layout a few definitions. Below, we use N to denote the number of qubits in a circuit, t the number of types of quantum gates in the search space, and n_t the total number of quantum primitive gates in one circuit.

3.1. Search space for QAS

We adopt two representations, list representation and image representation, to denote each candidate circuit. The two representations are utilized in the sampling strategy and predictor evaluation of QAS, respectively.

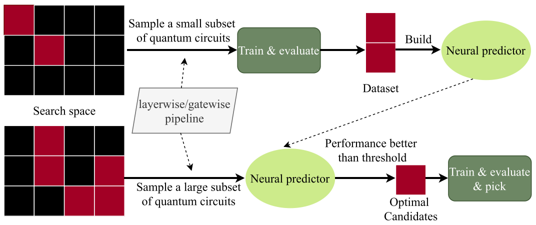

Sample a small subset of quantum circuits
Train & evaluate
Build
Dataset
Neural predictor
Search space
layerwise/gatewise pipeline
Performance better than threshold
Sample a large subset of quantum circuits
Neural predictor
Optimal Candidates
Train & evaluate & pick

Figure 1. The workflow of predictor based QAS. There are two phases of QAS. At phase I (the upper row), we generate the dataset of quantum circuits and train a neural predictor. At phase II (the lower row), we utilize the trained predictor to filter a large number of quantum circuit candidates where only a small fraction of them with predicted performance better than a threshold is further evaluated from the scratch. The ansatz with the best performance is picked as the final result of QAS.

The two representations can both represent the quantum circuit structure faithfully, and the choice is from engineering considerations. On the one hand, to generate the possible circuit structure in random search, list representation is simpler to manipulate and yield very diverse circuit structures. On the other hand, the input for the neural predictor must be a tensor, in which scenario, the image representation is more suitable. The list representation is used to generate circuit candidates and thus define the search space. It comes with a strict syntax delineating a quantum circuit in terms of a sequence of applied quantum gates. The list is a set of tuples, and each tuple encodes a quantum gate (referenced to a given gate set) and positional information (i.e. qubits numbered in a certain way). For instance, tuple $(3,1,2)$ indicates a two-qubit quantum gate of the 3rd type acting on the first and second qubit in the circuit. To reconstruct a corresponding quantum circuit from a list, the gates should be sequentially placed onto an initially empty circuit, in the order of the given tuple sequence. Note that a circuit ansatz may have multiple list representations. One possibility is that the order of gate placements can be freely exchanged as long as gates encoded by the two tuples commute. This syntax inherently gives a legitimate search space for circuit ansatz. In this work, sampled circuits are generated in this list representation.

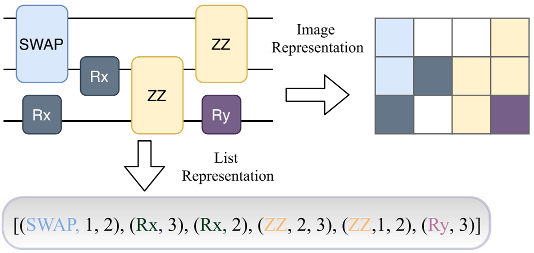
We choose different primitives for different problems. Some common examples include non-parameterized gates such as Hadamard gate H; single-qubit gates with trainable parameters such as rotation gate $\mathrm{Rx} = e^{-i\theta X/2}$ and, similarly, for Ry, Rz, where X, Y, Z are corresponding Pauli matrices. We also include parameterized two-qubit gates such as $\mathrm{XX} = e^{-i\theta X_1X_2/2}$ with counterparts for YY, ZZ gate, as well as parameterized SWAP gate $\mathrm{pSWAP} = e^{-i\theta \mathrm{SWAP}/2}$ where $\mathrm{SWAP} = I_1I_2 + X_1X_2 + Y_1Y_2 + Z_1Z_2$.

Now, let us elaborate on the sampling strategy considered in this work. A naïve method is by random sampling: after fixing the number of qubits and the total number of quantum gates in a circuit, one then randomly samples gate types and positions to form a list representation of the quantum circuit ansatz. This simple strategy has some undesirable features. Firstly, gate layouts for randomly sampled circuits are highly ‘chaotic’ and usually incur severe issues of barren plateau since the circuits behave like random unitaries drawn from the Haar measure. Secondly, with the total number of quantum primitive gates fixed, such randomly generated circuit ansatz is often deep as the arrangement of quantum gates is sparse. Lastly, random ansatz is not amenable to further utilization in the sense of generalization and transferability to problems of different sizes since there is no obvious pattern for extraction.

We devise two pipelines for circuit sampling that substantially alleviate the aforementioned concerns, which are gatewise generation and layerwise generation. In the first pipeline, we construct the circuit gate by gate, by specifying their positions and types, while in the second pipeline, we construct the circuit by iteratively adding half-layers. Namely, whenever we pick a type of quantum gate we apply it on the set of even qubits or odd qubits. Both pipelines further incorporate additional techniques such as hierarchical generation and gate correlation enforcement. See supplemental materials (available online at stacks.iop.org/MLST/2/045027/mmedia) for the details of these two sampling pipelines and the consideration on design of circuit search space.

3.2. Image representation of the quantum circuit

Apart from the list representation of quantum circuits, we have alluded to the image representation which is designed for training the neural predictor. To encode the circuit structure as input for an ML predictor model, we need a systematic way to represent circuit structures in the form of tensors. The strategy we invented is to transform a circuit into an image of multiple channels. The shape of the input tensor is $[\mathrm{depth}, \#\mathrm{qubits}, \#\mathrm{gate\ types}]$. The size of the figure is the number of qubits times the depth of the circuit, where the depth is the number of gate layers in a circuit. A similar representation of quantum circuits has been considered in [131]. For the circuit generation pipelines considered in this work, the total number of



[(SWAP, 1, 2), (Rx, 3), (Rx, 2), (ZZ, 2, 3), (ZZ,1, 2), (Ry, 3)]

Figure 2. List representation and image representation for circuit ansatz, $N = 3$, $D = 4$, $n_t = 6$.

quantum gates in a circuit n_t is fixed, but different candidate circuits tend to have incompatible circuit depth. Therefore, we need to set a max depth cutoff as D. All circuits with depth less than D are zero-padded up to D columns, and all circuits with depth more than D are simply retracted. In other words, we only process candidate circuits with n_t quantum gates and limited to a depth of D. Figure 2 gives an example of both the list representation and image representation for an $(N = 3, D = 4)$ quantum circuit.

For simplicity of presentation, we impose the following two assumptions on the search space: (a) the two-qubit gates are restricted to act on adjacent qubits, and (b) all two-qubit gate primitives in our work are symmetric. Under such assumptions, this image representation maintains a one-to-one mapping with the actual quantum circuit architecture. These two assumptions are not intrinsic limits of our general method and the image representation is easy to extend to maintain such one-to-one correspondence while the two assumptions are relaxed (see supplemental materials for the details).

In the proposed QAS workflow, one may further consider different architectures of ML predictors. If an recurrent neural network (RNN) based model is used, then we treat the depth dimension as the time dimension while the dimension of qubits and gate types is flattened out as an input vector for each time slice of such RNN based predictor.

3.3. The architecture of predictor model

We have tried various canonical neural networks, including multilayer perceptron, convolution neural network (CNN), and long short-term memory (LSTM), as the neural predictors to evaluate circuits in the proposed QAS workflow. In general, we find LSTM based RNN performs better than others in terms of predicting the fitness of circuit ansatz. See figure 3 for a schematic of the RNN neural predictor used in this work.

In some VQA problems, good circuit ansatzes are dense while bad circuit ansatz has a very wide range of fitness scores as a long tail distribution. Such a distribution is hard to fit with one regression model, and we adopt the strategy of two-stage classification for screening circuits. Firstly, a CNN based binary classification model is trained to differentiate between good and bad circuits for a task. Only good candidates are further fed into an RNN based regression model for a more fine-grained evaluation. Such regression model is only trained with good ansatz.

3.4. Training neural predictors

To train the neural predictor, we need to prepare a dataset composed of a circuit structure and its performance according to a task-specific evaluation metric, e.g. estimated energy in VQE simulation or validation accuracy for QML. It is important that the training of such a predictor model is sample efficient, i.e. a small number of training data points should allow the predictor to deliver an acceptable accuracy. Otherwise, predictor based QAS is not desired as it already takes an enormous amount of time and resources to process the training dataset. In our experiments, several hundreds of data pairs are in general enough to make QAS workflow a success for certain important applications. Of course, more training data can boost the predictor's accuracy and, in turn, elevate the overall efficiency for architecture search. There is certainly a tradeoff between the efforts to prepare the training dataset (for neural predictors) and the search efficiency in a later stage of a QAS workflow.

While preparing the training dataset, it is desired to evaluate each circuit multiple times with different initialization of parameters. This is because of an infamous fact, the energy landscape of typical cost functions for quantum circuits is often decorated with a plethora of local minima. Therefore, one should use the minimum loss of multiple runs as the training label for the neural predictor. There are additional benefits to run the same circuit multiple times. For instance, one may train another regression model to predict the standard deviation of the losses for multiple optimization runs of a quantum circuit. This frustration indicator gives us a hint of whether a candidate circuit can be consistently and easily trained for the task at

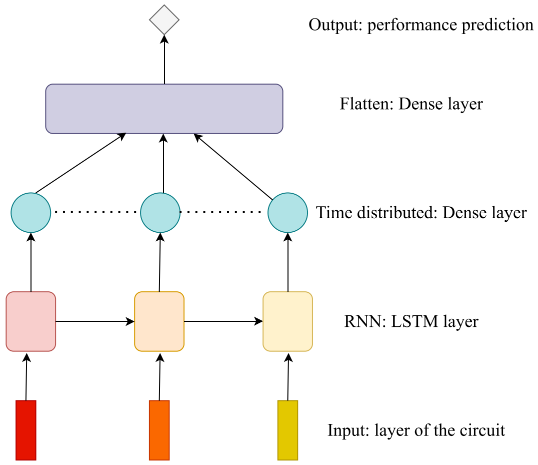

Output: performance prediction
Flatten: Dense layer
Time distributed: Dense layer
RNN: LSTM layer
Input: layer of the circuit

Figure 3. The schematic RNN neural architecture for the predictor of QAS. Layers of the circuit image representation are fed into the network as different time steps. The information processing flow in such a network is similar to the real quantum dynamics on the quantum circuit.

hand. A prediction of large standard deviation implies that the candidate circuit may suffer from a more ragged energy landscape as well as possible issues of barren plateaus. In principle, we can train a multi-task neural predictor that not only guide us for the circuit with a better potential to perform well but also easier to train from scratch.

Both mean squared loss and mean absolute loss are tried in the training of regression models, and they tend to give similar results. Adam optimizer is utilized for all the training in this work. Batch size is 32 or 64 in most of the training. Dropout layers with high dropout rates are heavily applied to avoid severe overfitting.

Furthermore, data augmentation can be applied to the image representation of quantum circuits. In many VQA problems, the final measurement observable is independent on the order of qubits, where a permutation on the qubits leaves the final result unchanged. Strictly speaking, in our search space there is no qubit permutation symmetry since two-qubit gates are only defined on the neighboring qubits and a random permutation may leave the predefined search space. But if we ignore this subtlety, input permutation on the qubits is still helpful to avoid overfitting as it essentially creates $N!$ times more data than the original input.

It is worth noting that the trained neural predictors often do not perform well in terms of conventional ML evaluation metrics. The training tends to overfit even with networks of fewer parameters and large dropout ratios. However, as we will show in the Results section, such predictors are actually good enough to greatly improve the search efficiency of QAS.

3.5. QAS workflow

Once the neural predictor is trained, we randomly sample a large number of quantum circuits according to the generation pipelines we introduce. Only circuits that pass the predictor screening (a tiny fraction of all sampled) will be actually tested to verify their fitness. In the end, one should pick a candidate circuit of best fitness for the current task. The entire workflow of neural predictor based QAS is succinctly summarized in figure 1.

Such neural predictor based evaluation policy can be combined with more sophisticated sampling strategies to further improve search efficiency. For example, the neural predictor can be further trained with additional data collected during the Phase II of QAS workflow. One can then screen more circuits with help from a refined predictor. This fine-tuning of the neural predictor can go a few rounds iteratively and should be beneficial for finding a really strong candidate circuit. Moreover, genetic transformation or Bayesian optimization can be used as sampling strategy in the Phase II of QAS to accelerate the search. Contrary to the plain random sampling utilized in this work, these advanced sampling strategies should help as in high-throughput virtual screenings of molecules and materials. Furthermore, neural predictors combined with a VAE setup can be exploited to directly search for strong candidate circuits with gradient-based method in the continuous latent space. We leave these interesting possibilities to extend predictor based QAS workflow as promising directions to explore in future work.

3.6. Transferability of optimal quantum ansatz

It is highly desirable that an optimal ansatz identified by QAS supports a transfer capacity since direct QAS on large systems is prohibitively difficult. For each optimal quantum circuit selected from QAS, we then test

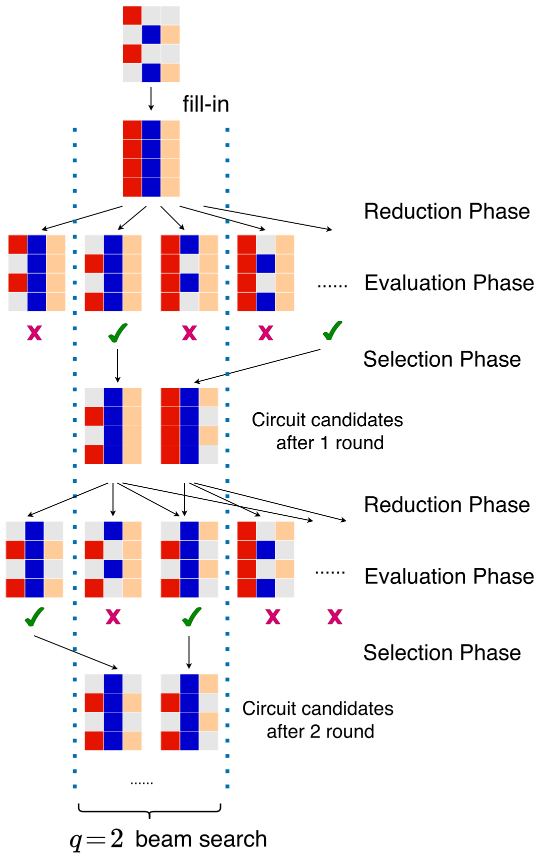


Figure 4. Schematic workflow of transferring optimal quantum ansatz. We first fill in the layerwise generated circuit structure and then do reductions by beam search ($q = 2$ in this example). The circuit structure is shown as image representation, where the light grey square indicates there is no quantum gate. The circuit in this example has $N = 4$ qubits and depth $D = 3$. Two steps of the beam search are shown and more steps are possible as long as the fitness in the evaluation phase is above the predefined threshold.

whether such circuit ansatz can be transferred to similar problems involving a larger number of qubits while maintaining state-of-the-art performance. Quantum circuits generated by the layerwise pipeline can be straightforwardly adapted to work on larger systems by simply extending the range of each gate to cover either all odd or even qubits in a larger circuit. However, such a direct transfer of the quantum ansatz is not rooted in rigorous theoretical analysis and may suffer from a huge performance drop.

In order to find a good ansatz on a large size system with the knowledge gained from QAS on small systems, we develop a beam search based method to accelerate the search for appropriate ansatz for large systems at a low extra overhead. Since quantum circuits generated by the layerwise pipeline always group the same quantum gate acting on either even or odd qubits in one layer, we can first fill the circuit by extending each quantum gate on all qubits (half-layer → full layer). Such 'fill-in' quantum circuits are, in general, very good candidates with great fitness for larger size problems. This comes at the price, however, as the new quantum circuit costs twice the quantum resource as the original circuit with half-layers. To reduce the number of gates while maintaining the performance above a threshold, we use a beam search scheme [51] to find simpler circuit structures with respect to a given 'fill-in' circuit. There are three phases at each step of the beam search, reduction phase, evaluation phase, and selection phase. In the reduction phase, we reduce quantum gates of one half-layer and every possible reduction ($2\,D$ in total) is generated. In the evaluation phase, these circuit candidates with reduction are evaluated for the fitness. Since these quantum structures can be viewed as subcircuits of the 'fill-in' one, the weights (trainable parameters) can be directly inherited and only some light fine-tuning is required to get a reasonably accurate evaluation. In the selection phase, only top-q circuits in terms of fitness are kept in the queue ($q = 1$ is reduced to greedy search algorithm). This iterative pruning of circuit structures ends when no further reduction on quantum gates is possible without compromising the expected performance. Finally, we note that this technique of beam search can also incorporate mutations or transformations of quantum circuits in addition to eliminations of quantum gates. In this augmented version of beam search, our transfer protocol operates like an evolutionary algorithm for finding suitable circuits for large-size problems. See figure 4 for the workflow schematic of such 'fill-in + beam search' protocol.

4. Results

In this section, we present the main results of neural predictor based QAS on two common VQA tasks: (a) supervised learning task for binary classification on the fashion-MNIST dataset and (b) quantum simulation to estimate the ground state energy of the transverse field Ising model (TFIM).

4.1. Quantum machine learning on classification of fashion-MNIST

4.1.1. Setup

We train a PQC as an ML model for a binary classification task on a commonly used benchmark in the ML community, and QAS is utilized to identify a suitable circuit architecture (conforming to specified constraints) that can attain rather high accuracy on the validation set. We choose fashion-MNIST [132], the dataset of clothes images, as the benchmark because it is more challenging than MNIST, the dataset of handwritten digits, commonly tested in QML literature. We only use the data labeled with T-shirt/top(0) and Dress(3) in order to focus on a binary classification problem instead of a multiple categorical classification. As demonstrated in [27], QML could perform better than the classical counterparts when the training dataset is small. We, hence, select only 500 datapoints from fashion-MNIST for the training purpose, and select another 500 datapoints for validation. Each 28×28 image of clothes is padding and flatten to a 1024-dimensional vector which is further encoded into a 10-qubit wavefunction using amplitude encoding. Our quantum ansatz is hence defined in the search space with qubit number $N = 10$, the number of quantum primitive gates $n_t = 50$, and circuit depth cutoff $D = 10$. We focus only on quantum ansatz sampled from the layerwise pipeline in this QML experiment. The primitives of quantum gates include $t = 7$ types in total: they are parameterized Rx, Ry, Rz, XX, YY, ZZ, and pSWAP gate.

Inspired from the recent proposal of classical shadow with random measurements [133, 134], we introduce a classical post-processing module to make the actual prediction based on features extracted from random measurements on the proposed quantum circuit. This 'random measurement' is implemented with one extra layer of Rx gate, appended to the end of the ansatz-generation circuit. Classical bits of information are then extracted with measurements of Pauli Z operator for each qubit as $\langle Z_i \rangle$. The collected classical information is subsequently fed into a classical dense layer as $y_{\text{pred}} = \sigma(\sum_{i=1}^{N} k_i \langle Z_i \rangle + b)$, where k_i and b are classical trainable parameters and $\sigma(x) = \frac{10}{1+e^{-x}}$ is a scaled sigmoid function. Finally, the mean squared loss between y_{pred} and y_{true} (the ground truth) is minimized with a simple gradient descent approach, where the weights of the quantum circuit and the classical post processing module are jointly optimized. Once the training of the QML model converges, we take the classification accuracy on the validation set as the evaluation metric.

Next, we discuss details of preparing the training dataset for neural predictors. For this QML benchmark, we adopt a loose converge condition in the phase I of QAS since the validation accuracy is known to be strongly correlated with the accuracy on the early stop. By using the early stop strategy, we may evaluate a circuit's prediction accuracy (for the fashion-MNIST data) without too many training steps. Furthermore, we decide to train each circuit only once, because it is rather time consuming to independently re-train a QML model multiple times. While we run the risk of getting a suboptimal estimate of a circuit's performance, we remind that our ultimate goal is to run a high-throughput screening of many quantum circuits in the phase II of a QAS workflow. Candidate circuits that pass the predictor-based screening still have to be experimentally trained and verified for their actual performances. As long as some optimal circuit architectures are detected and passed through the screening, a QAS is deemed successful. According to this perspective, building a highly accurate neural predictor is certainly desirable but not strictly indispensable.

We collect 300 datapoints for training the RNN based neural predictor. Each datapoint comprises a quantum circuit and its corresponding validation accuracy on the fashion-MINST as the label. The regression result is shown in figure 5. The performance can be evaluated by R^2 score of the linear regression which is around 0.71 on the validation dataset.

4.1.2. QAS result

The ultimate test for the QAS is how well the predictor-based screening performs in phase II. At this stage, we randomly generate quantum circuits from the layerwise pipeline, and predict their accuracy with the trained predictors. We only retain circuits with a predicted accuracy larger than 0.89 for further training and verification of their true accuracy in experiments. For 30 000 random circuits, only 195 (0.65%) are kept and proceed to the actual testings. The comparisons between the true accuracy of quantum circuits from random search (as the training dataset for the predictor) and the counterpart from QAS filtered by the neural predictor is summarized in figure 6. As shown, the optimal ansatz found by QAS has validation accuracy above 0.92, which is significantly larger than the best result seen in the training set (all lower than 0.9). The optimal quantum circuit recommended by QAS has a layerwise layout as YY, ZZ-odd, Rz-odd, YY-even,

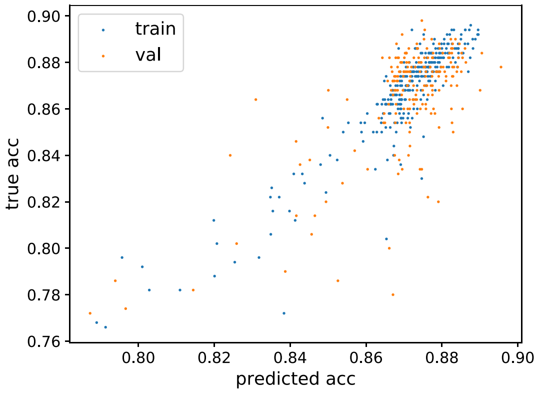


Figure 5. The regression performance of the trained neural predictor on training and validation set of quantum ansatz dataset.

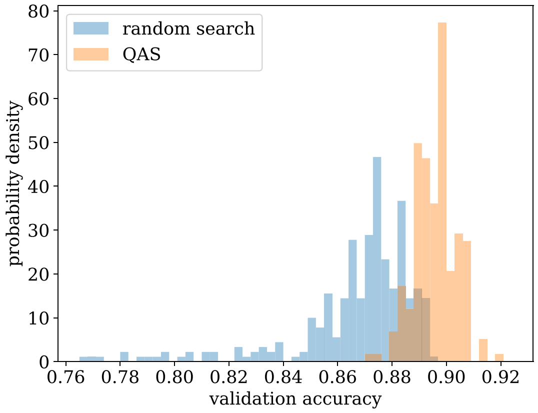


Figure 6. Validation accuracy histogram between quantum ansatz from random search and neural predictor based QAS in our work. The performance in terms of quantum machine learning classification accuracy is greatly improved in general for the set from QAS.

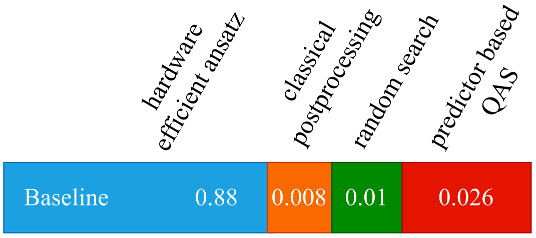


Figure 7. Validation accuracy on fashion-MNIST classification task from different contributing factors. The baseline (blue) accuracy is achieved by hardware efficient ansatz in the layer form of Rx, Ry, ZZ, Rx, Ry, ZZ. By attaching the classical post processing part from random measurements, the accuracy of hardware efficient ansatz gets improved to 0.888 (orange). With random search on 500 quantum circuit candidates, the best of them gives accuracy 0.898 (green). Further QAS via the neural predictor trained from random search data record the best accuracy of 0.924 for training on only $O(100)$ quantum circuit candidates (red).

Rz-even, ZZ-even, SWAP (see supplemental materials for the layered ansatz notation). It is interesting to observe the generalizability of such neural predictor based approach. As the predictor is only trained with suboptimal quantum circuits with accuracy less than 0.9, it can still single out quantum circuits with accuracy better than any one it has seen during the training.

We further compare QAS-screened QML model against a baseline established by a QML built with the conventional hardware efficient ansatz. The details are given in figure 7. As seen, both classical post processing and QAS contribute to the improved accuracy.

4.2. Variational quantum eigensolver for transverse field Ising model

4.2.1. Model

Next, we investigate how predictor based QAS may help to identify an efficient circuit for representing the ground state of a many-body Hamiltonian in a VQE simulation. We consider an $N = 6$ TFIM model with periodic boundary condition for illustration, which is one of the most representative models in many-body physics. The Hamiltonian is given by $H = \sum_i Z_i Z_{i+1} + X_i$, and this system can be modeled with six qubits, whose exact ground state energy is $E_0 = -7.7274066$. The evaluation metric in this case is simply the best energy estimation from VQE optimization.

4.2.2. QAOA baseline

We first give a QAOA-inspired circuit as the baseline for this VQE simulation. This baseline circuit generalizes the QAOA structure by allowing parameters to be independently tunable intralayer. More precisely, the ansatz wavefunction reads $\prod_p \prod_i (e^{i\phi_{pi} X_i} e^{i\theta_{pi} Z_i Z_{i+1}}) \prod_i H_i |0\rangle$, where p is the number of layers for the ansatz, all ϕ, θ are trainable parameters and H_i is Hadamard gate on qubit i. In the following analysis, we compare results from QAS-designed circuit against such baselines derived from this QAOA-inspired circuit with $p = 1, 2, 3$, respectively. For the record, these three baselines estimate the TFIM ground-state energy to be -7.24264, -7.4641, -7.7274066, respectively. In comparison to the true energy, $p = 3$ ansatz with 42 quantum gates and 36 parameters can fully represent the ground state of the $N = 6$ TFIM system.

4.2.3. Setup

Since obtaining optimal weights for a VQE circuit usually takes less time than that for a QML task requiring a large dataset, we train each quantum circuit for 10 independent runs with different random initializations to avoid local minimum in the estimation of the ground-state energy. These independent training can be cast into the batch dimension with the help of QML framework [135] by clever design which enables fast optimization over independent runs in parallel. The search space for this VQE task is confined to a qubit number $N = 6$, a total number of quantum primitive gates $n_t = 36$ with depth cutoff $D = 10$. Candidate circuits are sampled from both layerwise and gatewise generation pipelines with equal contribution. The primitive set of quantum gates include H, Rx, Ry, Rz, XX, YY, and ZZ gate this time. Among these gates, the Hadamard gate H is the only type without a tunable parameter. For the layerwise pipeline, we set 30% of all generated circuits to begin with a layer of the Hadamard gate applied to all qubits, as starting from the state $|+\rangle$ may help VQE to find a better approximation to the ground state.

We adopt the two-stage screening for this VQE investigation. The rationale is that the energy distribution of the TFIM model is not smooth across a wide region in the search space; therefore, a single high-quality predictor is difficult to train. To overcome this problem, we resort to using two predictors as described in Method section. First, we use a CNN based binary classification model to quickly rule out inappropriate circuits covering a wide variety of rather arbitrary circuit structures. The 'good' circuits with limited variety are then screened again with an RNN-based regression model to evaluate their performance more precisely. In this case, we use $\epsilon = (E - E_0)/14 \in [0, 1)$, the normalized deviation of the estimated ground-state energy, as the predicted label.

We again randomly pick and optimize 300 independent quantum circuits to build the training datasets for the two neural predictors. Figure 8 shows the distribution of converged energy (error ratio ϵ) for the dataset of 300 circuits. Such a distorted distribution manifests the source of difficulty to rely on a single regression model to directly pick out top-performing circuits from the entire candidate pool. Rather, in our two-stage screening setups, the regression model is only expected to provide an accurate characterization of potentially good circuit candidates with limited variety.

For the CNN based classification model, we adopt data augmentation by applying random permutations on qubits since TFIM is translationally invariant and the energy is agnostic to the order of qubits. In addition, multiple-layer convolution with dilation, batch normalization on qubit dimension, dropout, ELU activation, and L_2 regularization on weights are all utilized in the classification model.

The training dataset we used only contains 300 quantum circuits and their corresponding error ratios obtained from VQE optimizations. When one is limited to such a small training set, it could be crucial how the circuits are selected to form the circuit pool for a particular problem. Prior knowledge (such as insights from physics) could prove beneficial at this stage. For instance, only circuits conforming to certain symmetry are generated. However, for the present study, we want to emphasize the universality of this predictor-based QAS and simply do a random sampling of circuits according to the two sampling pipelines. With such a scarcity of training data, the neural predictors actually do not perform particularly well according to the traditional metrics for ML model evaluation. Nevertheless, we remind that a complete QAS workflow operates more like a high-throughput screening with the ultimate goal of discovering one or a few top-performing circuits as opposed to making highly accurate predictions for all circuits. Despite this focus

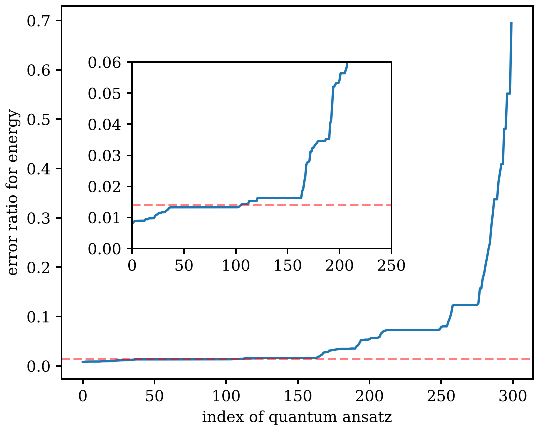


Figure 8. Sorted error ratio for VQE energy from the training dataset of quantum ansatz. The red dash line (0.014) is our threshold for the CNN based classification model at the first stage, i.e. we train the classification model to determine good candidates if their error ratio is less than 0.014. The inset is the zoom-in for good quantum ansatz candidates part.

being different from the standard ML, we still have to deal with some non-trivial challenges shared by many ML tasks. For the binary classification, we face a highly imbalanced classification with a trade-off between precision and recall. In this case, we prefer a model with higher precision to recall, as the goal is to efficiently filter a large search space. Specifically, our trained CNN based classification model has a precision around 0.7 and a recall around 0.47.

4.2.4. QAS result

With the two-stage predictor based screening in place, we filter a large number of circuits generated via either gatewise or layerwise pipelines. The filter thresholds for the two models are set to 0.85 and 0.005, respectively. We only keep the most confident candidates for further VQE evaluations. At the first stage, only quantum circuits with predicted value larger than 0.85 instead of 0.5 are kept as promising candidates for further evaluation. At the second stage, only candidates with predicted error ratio less than 0.005 are recommended for experimental verification (i.e. going through standard VQE optimizations). Remarkably, our RNN-based regression model can give outputs with predicted error ratio less than 0.005 when the smallest value in its training dataset 0.00789 is larger than this threshold.

We randomly sample 50 000 quantum circuits, and only 626 (1.25%) pass the two-stage screening and are evaluated with the VQE optimization. Among these final candidates, five circuits give an energy less than -7.7 and we call them optimal ansatz. This is an interesting result as the training dataset of 300 quantum circuits does not feature such good circuits—the best energy estimation in our training dataset is only -7.61694—yet predictor trained with suboptimal circuits finds optimal ones. This observation demonstrates the effectiveness of our few-data trained predictors to screen unseen structures in a QAS context. In this case, the search efficiency for the predictor-based QAS workflow is around 1%, i.e. one out of every 100 quantum circuits recommended for VQE experiments are optimal (in the sense of VQE energy less than -7.7). Without the neural predictor as the filter, random search efficiency is around $1/2000$ following our search space pipelines according to a large number of numerical simulations. In other words, on average, predictor-based QAS only needs to conduct 100 independent VQEs before discovering an optimal candidate while naive random search requires 2000 circuits evaluation before encountering a good candidate in our pre-defined search space. This is a 20 times boost of search efficiency. Figure 9 summarizes the gain brought by the predictor based QAS compared to the random search.

Since in NAS or QAS, a fair efficiency comparison between different approaches is of high importance, we briefly discuss on the possibility of comparison between our predictor based approach and better QAS alternatives than random search baseline. There are two relatively independent ingredients in QAS, search strategy and evaluation method, that contribute to the final performance of QAS. Most QAS approaches focus on designing a powerful search strategy, such as RL or genetic algorithm based setups. Instead, our approach proposed a new way to improve the evaluation method, i.e. we replace the plain evaluation method (train then evaluate) with neural predictors. Specifically, the comparison of the 20 times acceleration above is from random search + plain evaluation method versus random search + neural predictor evaluation. Note that neural predictor evaluation can be seamlessly combined with other advanced search strategies, as they belong to different aspects of QAS. Therefore, another fair comparison, like RL-QAS + plain evaluation method versus RL-QAS + neural predictor evaluation, should achieve roughly the same level of acceleration

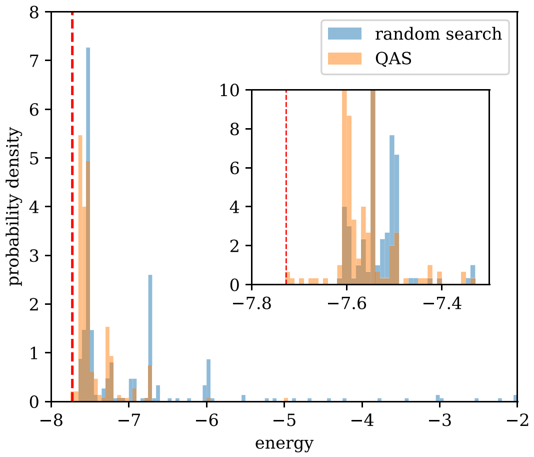


Figure 9. VQE energy histogram between quantum ansatz from random search and neural predictor based QAS. 300 samples from random search are also the training dataset of the predictor (blue). QAS can find quantum ansatz with lower energy on average and select several candidates with energy less than the best result from the training dataset. The red dash line is the exact ground state energy given by exact diagonalization. The optimal ansatz found by QAS matches the ground truth.

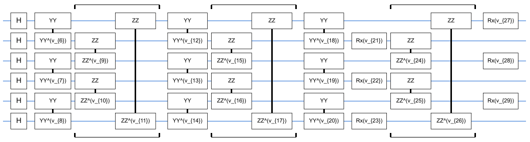


Figure 10. The optimal quantum ansatz found by neural predictor on $n_t = 30$ search space. v_{i} are trainable parameters for such VQE ansatz. The layerwise ansatz can be summarized as H, YY-odd, ZZ-even, YY-odd, ZZ-even, YY-odd, Rx-even, ZZ-even, Rx-odd following our convention on layerwise ansatz.

since the search strategy keeps consistent. On the contrary, the comparison between better alternative search strategies + plain evaluation method versus random search baseline + neural predictor evaluation is less ideal, as we can easily incorporate these better search strategies into neural predictor based QAS.

4.2.5. Transfer of the predictor

Next, recall that our predictors are trained with circuits limited to $n_t = 36$ quantum gates, but it can be transferred to predict the fitness of quantum circuits consisting of a different number of quantum gates with zero tuning. For example, we know that QAOA-inspired ansatz with $p = 3$ and $n_t = 42$ gates can give the exact ground state. Although the total gate number is substantially changed compared to the instances in the training set ($n_t = 36$) for the predictors, the screening pipeline still works well. The classification predictor at the first stage gives the output 0.99979 and the regression predictor gives 0.004 as the predicted error ratio for the QAOA-inspired circuit with $p = 3$. In short, this optimal circuit successfully passes the two-stage screening without having to train these predictors with circuits comprising the same number of quantum gates.

The predictors are not only able to make reasonably accurate predictions for circuits having more quantum gates, but also ones with fewer gates too. For instance, we conduct a large scale search for circuits with $n_t = 30$ gates. 360 out of 100000 quantum circuit structures are screened, and three instances of them give VQE energy smaller than −7.7 with the best one being −7.7274. This is a highly nontrivial result, as QAOA ansatz with $p = 2$ and the same amount of quantum resources only gives energy −7.46. The optimal circuit structure we find by transferred predictor is displayed in figure 10.

4.2.6. Transferability of the optimal ansatz

Apart from the neural predictors, the QAS-designed quantum ansatz can also be transferred to guide the search for optimal circuits for similar but larger-sized problems involving more qubits in the layerwise search space. Such transferability is desired as it is time consuming to directly conduct QAS in a larger search space. Therefore, we propose that QAS on large systems should proceed in two steps. First, we run QAS on smaller systems as a proxy task, then we transfer the optimal ansatz by QAS to larger systems. We conduct such transfer experiments from the optimal ansatz in figure 10 to a larger TFIM model with $N = 10$ spins. The exact energy and QAOA baseline with $p = 3$ for $N = 10$ TFIM are −12.7849 and −12.56758, respectively.

We adopt the beam search approach developed in the Method section. By starting from a fully fill-in quantum circuit, we utilize the beam search to reduce potentially redundant quantum layers without compromising the final performance too much. By this approach, we obtain the following circuit structure: H-even, YY, ZZ, YY, ZZ-even, YY-odd, Rx-even. Such ansatz gives a VQE energy estimation of −12.634, significantly better than the QAOA baseline with $p = 3$ (45 trainable parameters in the transferred ansatz versus 60 trainable parameters in the QAOA-inspired circuit). If we further relax the optimal threshold, we obtain a circuit structure: H-even, YY, ZZ, YY Rx-even. This extremely compact ansatz containing 35 trainable parameters and 40 quantum primitive units has a VQE energy of −12.587, outperforming the more complex QAOA-inspired ansatz with $p = 3$. These numerical results strongly support our strategy to transfer optimal circuit structures to similar many-body simulation problems involving different numbers of qubits.

5. Discussions

There are various future directions to refine the proposed QAS workflow and to explore other novel strategies to discover optimal quantum circuits. One obvious possibility is to combine advanced sampling engines than the simple random search with our predictor based evaluation policy in QAS. For example, we may invoke evolutionary algorithm based sampling policy together with the learned predictor, which might further improve the search efficiency as witnessed in many high-throughput virtual screening studies. Next, we may extend the phase II of the current QAS workflow into a loop with multiple rounds of screening. Hence, the neural predictors can be iteratively updated and fine tuned as bathes of new data points become available within each round of QAS verification of proposed circuits. Moreover, weight sharing mechanism can be combined with the predictor approach to further speed up the preparation of the training dataset. Also, it is of great importance to further investigate transferability and propose more systematic transfer protocols for the optimal ansatz, since challenging problems are always large sized. Finally, other types of neural predictors might be helpful, such as fast indicators for quantum noise resilience [131] or frustration in the training energy landscape. These additional considerations may be particularly crucial to establishing non-trivial quantum advantages with VQA-based approaches in the NISQ era.

6. Conclusion

In this work, we introduced neural predictor as the evaluation policy for QAS. We demonstrate the effectiveness of predictor based QAS on various examples from VQE and QML. We find greatly improved search efficiency and new state-of-the-art quantum architectures for these VQA tasks. Besides, we show how the trained predictors, as well as the QAS-designed optimal ansatz, are capable of being transferred to a different ansatz search space or problems of different size, respectively.

Data availability statement

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

Acknowledgments

S-X Z would like to thank Zhou-Quan Wan for helpful discussions. S-X Z and H Y are supported in part by the NSFC under Grant No. 11825404. H Y is also supported in part by the MOSTC under Grant Nos. 2016YFA0301001 and 2018YFA0305604, and the Strategic Priority Research Program of Chinese Academy of Sciences under Grant No. XDB28000000.

ORCID iD

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

References

[1] Cerezo M, Arrasmith A, Babbush R, Benjamin S C, Endo S, Fujii K, McClean J R, Mitarai K, Yuan X, Cincio L, and Coles P J 2020 Variational quantum algorithms (arXiv:2012.09265)
[2] Bharti K, Cervera-Lierta A, Kyaw T H, Haug T, Alperin-Lea S, Anand A, Degroote M, Heimonen H, Kottmann J S, Menke T, Mok W-K, Sim S, Kwek L-C and Aspuru-Guzik A 2021 Noisy intermediate-scale quantum (NISQ) algorithms (arXiv:2101.08448)
[3] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[4] 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

[5] O'Malley P J J *et al* 2016 Scalable quantum simulation of molecular energies *Phys. Rev.* X **6** 031007

[6] 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

[7] Liu J-G, Zhang Y-H, Wan Y and Wang L 2019 Variational quantum eigensolver with fewer qubits *Phys. Rev. Res.* **1** 023025

[8] McArdle S, Endo S, Aspuru-Guzik A, Benjamin S C and Yuan X 2020 Quantum computational chemistry *Rev. Mod. Phys.* **92** 015003

[9] 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

[10] Farhi E, Goldstone J and Gutmann S 2014 A quantum approximate optimization algorithm (arXiv:1411.4028)

[11] 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

[12] 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

[13] Arute F *et al* 2020 Quantum approximate optimization of non-planar graph problems on a planar superconducting processor (arXiv:2004.04197)

[14] McClean J R, Harrigan M P, Mohseni M, Rubin N C, Jiang Z, Boixo S, Smelyanskiy V N, Babbush R and Neven H 2020 Low depth mechanisms for quantum optimization (arXiv:2008.08615)

[15] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning *Nature* **549** 195

[16] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)

[17] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309

[18] Liu J-G and Wang L 2018 Differentiable learning of quantum circuit Born machines *Phys. Rev.* A **98** 062324

[19] Lloyd S and Weedbrook C 2018 Quantum generative adversarial learning *Phys. Rev. Lett.* **121** 040502

[20] 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)

[21] Havlíček V, Córcoles A D, Temme K, Harrow A W, Kandala A, Chow J M and Gambetta J M 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209

[22] Schuld M and Killoran N 2019 Quantum machine learning in feature Hilbert spaces *Phys. Rev. Lett.* **122** 040504

[23] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001

[24] 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

[25] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273

[26] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 Circuit-centric quantum classifiers *Phys. Rev.* A **101** 032308

[27] Huang H-Y, Broughton M, Mohseni M, Babbush R, Boixo S, Neven H, and McClean J R 2020 Power of data in quantum machine learning (arXiv:2011.01938)

[28] Kandala A, Mezzacapo A, Temme K, Takita M, Brink M, Chow J M and Gambetta J M 2017 Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets *Nature* **549** 242

[29] Wecker D, Hastings M B and Troyer M 2015 Progress towards practical quantum variational algorithms *Phys. Rev.* A **92** 042303

[30] Sim S, Johnson P D and Aspuru-Guzik A 2019 Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms *Adv. Quantum Technol.* **2** 1900070

[31] 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)

[32] Akshay V, Philathong H, Morales M E S and Biamonte J D 2020 Reachability deficits in quantum approximate optimization *Phys. Rev. Lett.* **124** 090504

[33] Rasmussen S E, Loft N J S, Bækkegaard T, Kues M, and Zinner N T 2020 Single-qubit rotations in parameterized quantum circuits (arXiv:2005.13548)

[34] Akshay V, Philathong H, Zacharov I, and Biamonte J 2020 Reachability deficits implicit in Google's quantum approximate optimization of graph problems (arXiv:2007.09148)

[35] 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)

[36] Elsken T, Metzen J H and Hutter F 2018 Neural architecture search: a survey *J. Mach. Learn. Res.* **20** 1

[37] Wistuba M, Rawat A, and Pedapati T, 2019 A survey on neural architecture search (arXiv:1905.01392)

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

[39] Zhang S-X, Hsieh C-Y, Zhang S, and Yao H 2020 Differentiable quantum architecture search (arXiv:2010.08561)

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

[41] 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

[42] Cincio L, Subaşı Y, Sornborger A T, Coles P J, Sotnikov O M, Cincio L, Subaşı Y, Sornborger A T and Coles P J 2018 Learning the quantum algorithm for state overlap *New J. Phys.* **20** 113022

[43] Rattew A G, Hu S, Pistoia M, Chen R, and Wood S, Domain-agnostic A 2019 Noise-resistant, hardware-efficient evolutionary variational quantum Eigensolver (arXiv:1910.09694)

[44] Cincio L, Rudinger K, Sarovar M and Coles P J 2020 Machine learning of noise-resilient quantum circuits (arXiv:2007.01210)

[45] 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)

[46] Lu Z, Shen P-X, and Deng D-L 2020 Markovian quantum neuroevolution for machine learning (arXiv:2012.15131)

[47] Fösel T, Tighineanu P, Weiss T and Marquardt F 2018 Reinforcement learning with neural networks for quantum feedback *Phys. Rev.* X **8** 31084

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

[49] Ostaszewski M, Grant E, and Benedetti M 2019 Quantum circuit structure learning (arXiv:1905.09692)

[50] Tang H L, Shkolnikov V O, Barron G S, Grimsley H R, Mayhall N J, Barnes E, and Economou S E 2019 Qubit-ADAPT-VQE: an adaptive algorithm for constructing hardware-efficient ansatze on a quantum processor (arXiv:1911.10205)
[51] 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
[52] Bender G, Kindermans P J, Zoph B, Vasudevan V and Le Q 2018 Understanding and simplifying one-shot architecture search *ICML* 883
[53] Luo R, Qin T, and Chen E 2019 Balanced one-shot neural architecture optimization (arXiv:1909.10815)
[54] Guo Z, Zhang X, Mu H, Heng W, Liu Z, Wei Y and Sun J 2020 Single path one-shot neural architecture search with uniform sampling *ECCV* 544
[55] Liu H, Simonyan K and Yang Y 2019 DARTS: differentiable architecture search *ICLR*
[56] 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)
[57] Deng B, Yan J, and Lin D 2017 Peephole: predicting network performance before training (arXiv:1712.03351)
[58] Liu C, Zoph B, Neumann M, Shlens J, Hua W, Li L J, Fei-Fei L, Yuille A, Huang J, and Murphy K 2017 Progressive neural architecture search (arXiv:1712.00559)
[59] Dai X *et al* 2019 ChamNet: towards efficient network design through platform-aware model adaptation *CVPR*
[60] Wang L, Zhao Y, Jinnai Y, and Fonseca R 2019 AlphaX: eXploring neural architectures with deep neural networks and Monte Carlo tree search (arXiv:1903.11059)
[61] White C, Ai R and Neiswanger W 2019 Neural architecture search via Bayesian optimization with a neural network model *NeurIPS*
[62] Shi H, Pi R, Xu H, Li Z, Kwok J T and Zhang T 2019 Bridging the Gap between sample-based and one-shot neural architecture search with BONAS *NeurIPS*
[63] Wen W, Liu H, Chen Y, Li H, Bender G and Kindermans P J 2020 Neural Predictor for Neural Architecture Search *Lect. Notes Comput. Sci. Eng.* **12374** 660
[64] Ning X, Zheng Y, Zhao T, Wang Y, and Yang H 2020 A generic graph-based neural architecture encoding scheme for predictor-based NAS (arXiv:2004.01899)
[65] Chen X, Xie L, Wu J, Wei L, Xu Y, and Tian Q 2020 Fitting the search space of weight-sharing nas with graph convolutional networks (arXiv:2004.08423)
[66] White C, Neiswanger W, Nolen S, and Savani Y 2020 A study on encodings for neural architecture search (arXiv:2007.04965)
[67] Lukasik J, Friede D, Stuckenschmidt H, and Keuper M 2020 Neural architecture performance prediction using graph neural networks (arXiv:2010.10024)
[68] Dudziak Ł, Chau T, Abdelfattah M S, Lee R, Kim H, and Lane N D 2020 BRP-NAS: prediction-based NAS using GCNs (arXiv:2007.08668)
[69] Mauch L, Tiedemann S, Garcia J A, Cong B N, Yoshiyama K, Cardinaux F, and Kemp T 2020 Efficient sampling for predictor-based neural architecture search (arXiv:2011.12043)
[70] Li L and Talwalkar A 2019 Random search and reproducibility for neural architecture search (arXiv:1902.07638)
[71] Huang S, Li X, Cheng Z-Q, Zhang Z, and Hauptmann A 2018 GNAS: a greedy neural architecture search method for multi-attribute learning (arXiv:1804.06964)
[72] White C, Nolen S, and Savani Y 2020 Local search is state of the art for NAS benchmarks (arXiv:2005.02960)
[73] Li G, Qian G, Delgadillo I C, Müller M, Thabet A, and Ghanem B 2019 SGAS: sequential greedy architecture search (arXiv:1912.00195)
[74] 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 *ICML* 2902
[75] Xie L and Yuille A 2017 Genetic CNN *ICCV* 1388
[76] Liu H, Simonyan K, Vinyals O, Fernando C and Kavukcuoglu K 2018 Hierarchical representations for efficient architecture search *ICLR*
[77] Real E, Aggarwal A, Huang Y and Le Q V 2019 Regularized evolution for image classifier architecture search *AAAI* 4780
[78] Stanley K O, Clune J, Lehman J and Miikkulainen R 2019 Designing neural networks through neuroevolution *Nat. Mach. Intell.* **1** 24
[79] Zoph B and Le Q V 2017 Neural architecture search with reinforcement learning *ICLR*
[80] Baker B, Gupta O, Naik N and Raskar R 2017 Designing neural network architectures using reinforcement learning *ICLR*
[81] Cai H, Chen T, Zhang W, Yu Y and Wang J 2018 Efficient architecture search by network transformation *AAAI* 2787
[82] Zoph B, Vasudevan V, Shlens J and Le Q V 2018 Learning transferable architectures for scalable image recognition *CVPR* 8697
[83] Casale F P, Gordon J, and Fusi N 2019 Probabilistic neural architecture search (arXiv:1902.05116)
[84] Cai H, Zhu L and Han S 2019 Proxyless nas: direct neural architecture search on target task and hardware *ICLR*
[85] Xie S, Zheng H, Liu C and Lin L 2019 SNAS: stochastic neural architecture search *ICLR*
[86] Kingma D P and Welling M 2014 Auto-encoding variational Bayes *ICLR*
[87] Luo R, Tian F, Qin T, Chen E and Liu T Y 2018 Neural architecture optimization *NeurIPS* 7816
[88] Zhang M, Jiang S, Cui Z, Garnett R and Chen Y 2019 D-VAE: A variational autoencoder for directed acyclic graphs *NeurIPS*
[89] Friede D, Lukasik J, Stuckenschmidt H, and Keuper M 2019 A variational-sequential graph autoencoder for neural architecture performance prediction (arXiv:1912.05317)
[90] Cheng H P, Zhang T, Li S, Yan F, Li M, Chandra V, Li H, and Chen Y 2020 NASGEM: neural architecture search via graph embedding method (arXiv:2007.04452)
[91] Li J, Liu Y, Liu J, and Wang W 2020 Neural architecture optimization with graph VAE (arXiv:2006.10310)
[92] Zheng X, Wang P, Wang Q, and Shi Z 2020 Disentangled neural architecture search (arXiv:2009.13226)
[93] Tang Y, Wang Y, Xu Y, Chen H, Shi B, Xu C, Xu C, Tian Q and Xu C 2020 A semi-supervised assessor of neural architectures *CVPR* 1807
[94] Cao Y *et al* 2019 Quantum chemistry in the age of quantum computing *Chem. Rev.* **119** 10856
[95] Higgott O, Wang D and Brierley S 2019 Variational Quantum Computation of Excited States *Quantum* **3** 156
[96] Nakanishi K M, Mitarai K and Fujii K 2019 Subspace-search variational quantum eigensolver for excited states *Phys. Rev. Res.* **1** 033062
[97] Hsieh C Y, Sun Q, Zhang S and Lee C K 2021 Unitary-coupled restricted Boltzmann machine ansatz for quantum simulations *npj Quantum Inf.* **7** 19

[98] Li Y and Benjamin S C 2017 Efficient Variational Quantum Simulator Incorporating Active Error Minimization *Phys. Rev.* X **7** 021050

[99] Yuan X, Endo S, Zhao Q, Li Y and Benjamin S C 2019 Theory of variational quantum simulation *Quantum* **3** 191

[100] McArdle S, Jones T, Endo S, Li Y, Benjamin S C and Yuan X 2019 Variational ansatz-based quantum simulation of imaginary time evolution *npj Quantum Inf.* **5** 75

[101] Cîrstoiu C, Holmes Z, Iosue J, Cincio L, Coles P J and Sornborger A 2020 Variational fast forwarding for quantum simulation beyond the coherence time *npj Quantum Inf.* **6** 82

[102] Lin S-H, Dilip R, Green A G, Smith A, and Pollmann F 2020 Real- and imaginary-time evolution with compressed quantum circuits (arXiv:2008.10322)

[103] Lau J W Z, Bharti K, Haug T, and Kwek L C 2021 Quantum assisted simulation of time dependent Hamiltonians (arXiv:2101.07677)

[104] Endo S, Sun J, Li Y, Benjamin S C and Yuan X 2020 Variational quantum simulation of general processes *Phys. Rev. Lett.* **125** 010501

[105] Benedetti M, Fiorentini M, and Lubasch M 2020 Hardware-efficient variational quantum algorithms for time evolution (arXiv:2009.12361)

[106] Bharti K and Haug T 2020 Quantum assisted simulator (arXiv:2011.06911)

[107] Barison S, Vicentini F, and Carleo G 2021 An efficient quantum algorithm for the time evolution of parameterized circuits (arXiv:2101.04579)

[108] Liu J-G, Mao L, Zhang P, and Wang L 2019 Solving quantum statistical mechanics with variational autoregressive networks and quantum circuits (arXiv:1912.11381)

[109] Wang Y, Li G, and Wang X 2020 Variational quantum Gibbs state preparation with a truncated Taylor series (arXiv:2005.08797)

[110] Chowdhury A N, Low G H, and Wiebe N 2020 A variational quantum algorithm for preparing quantum Gibbs states (arXiv:2002.00055)

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

[112] Heya K, Suzuki Y, Nakamura Y, and Fujii K 2018 Variational quantum gate optimization (arXiv:1810.12745)

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

[114] Anschuetz E, Olson J, Aspuru-Guzik A and Cao Y 2019 *Lect. Notes Comput. Sci. Eng.* **11413** 74–85 LNCS

[115] Saxena A, Shukla A, and Pathak A 2020 A hybrid scheme for prime factorization and its experimental implementation using IBM quantum processor (arXiv:2009.05840)

[116] Karamlou A H, Simon W A, Katabarwa A, Scholten T L, Peropadre B, and Cao Y 2020 Analyzing the performance of variational quantum factoring on a superconducting quantum processor (arXiv:2012.07825)

[117] Huang H Y, Bharti K, and Rebentrost P 2019 Near-term quantum algorithms for linear systems of equations (arXiv:1909.07344)

[118] Xu X, Sun J, Endo S, Li Y, Benjamin S C, and Yuan X 2019 Variational algorithms for linear algebra (arXiv:1909.03898)

[119] Bravo-Prieto C, LaRose R, Cerezo M, Subasi Y, Cincio L, and Coles P J 2019 Variational quantum linear solver (arXiv:1909.05820)

[120] Kyriienko O, Paine A E, and Elfving V E 2020 Solving nonlinear differential equations with differentiable quantum circuits (arXiv:2011.10395)

[121] Lubasch M, Joo J, Moinier P, Kiffner M and Jaksch D 2020 Variational quantum algorithms for nonlinear problems *Phys. Rev.* A **101** 010301

[122] Yung M-H, Casanova J, Mezzacapo A, McClean J, Lamata L, Aspuru-Guzik A and Solano E 2015 From transistor to trapped-ion computers for quantum chemistry *Sci. Rep.* **4** 3589

[123] 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

[124] Wang S, Fontana E, Cerezo M, Sharma K, Sone A, Cincio L, and Coles P J 2020 Noise-induced Barren plateaus in variational quantum algorithms (arXiv:2007.14384)

[125] Temme K, Bravyi S and Gambetta J M 2017 Error mitigation for short-depth quantum circuits *Phys. Rev. Lett.* **119** 180509

[126] Grant E, Wossnig L, Ostaszewski M and Benedetti M 2019 An initialization strategy for addressing barren plateaus in parametrized quantum circuits *Quantum* **3** 214

[127] Patti T L, Najafi K, Gao X, and Yelin S F 2020 Entanglement devised Barren plateau mitigation (arXiv:2012.12658)

[128] Cerezo M, Sone A, Volkoff T, Cincio L, and Coles P J 2020 Cost-function-dependent barren plateaus in shallow quantum neural networks (arXiv:2001.00550)

[129] Franca D S and Garcia-Patron R 2020 Limitations of optimization algorithms on noisy quantum devices (arXiv:2009.05532)

[130] Pirhooshyaran M and Terlaky T 2020 Quantum circuit design search, (arXiv:2012.04046)

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

[132] Xiao H, Rasul K, and Vollgraf R 2017 Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms (arXiv:1708.07747)

[133] Huang H Y, Kueng R and Preskill J 2020 Predicting many properties of a quantum system from very few measurements *Nat. Phys.* **16** 1050

[134] Li G, Song Z, and Wang X 2020 VSQL: variational shadow quantum learning for classification (arXiv:2012.08288)

[135] Broughton M, Verdon G, McCourt T, Martinez A J, Yoo J H, Isakov S V, Massey P, Niu M Y, Halavati R, Peters E, Leib M, Skolik A, Streif M, Dollen D V, McClean J R, Boixo S, Bacon D, Ho A K, Neven H, and Mohseni M 2020 TensorFlow quantum: a software framework for quantum machine learning (arXiv:2003.02989)