Quantum autoencoders for communication-efficient quantum cloud computing

Yan Zhu\textsuperscript{1,2}, Ge Bai\textsuperscript{1,2}, Yuexuan Wang\textsuperscript{3}, Tongyang Li\textsuperscript{4,5,6}, and Giulio Chiribella\textsuperscript{1,7,8,9}

\textsuperscript{1}QICI Quantum Information and Computation Initiative, Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong
\textsuperscript{2}HKU-Oxford Joint Laboratory for Quantum Information and Computation
\textsuperscript{3}College of Computer Science and Technology, Zhejiang University, Hangzhou, China
\textsuperscript{4}Center on Frontiers of Computing Studies, Peking University
\textsuperscript{5}School of Computer Science, Peking University
\textsuperscript{6}Center for Theoretical Physics, Massachusetts Institute of Technology
\textsuperscript{7}Department of Computer Science, University of Oxford, Parks Road, Oxford OX1 3QD, United Kingdom
\textsuperscript{8}Perimeter Institute For Theoretical Physics, 31 Caroline Street North, Waterloo N2L 2Y5, Ontario, Canada
\textsuperscript{9}The University of Hong Kong Shenzhen Institute of Research and Innovation, Yuexing 2nd Rd Nanshan, Shenzhen 518057, China

In the model of quantum cloud computing, the server executes a computation on the quantum data provided by the client. In this scenario, it is important to reduce the amount of quantum communication between the client and the server. A possible approach is to transform the desired computation into a compressed version that acts on a smaller number of qubits, thereby reducing the amount of data exchanged between the client and the server. Here we propose quantum autoencoders for quantum gates (QAEGate) as a method for compressing quantum computations. We illustrate it in concrete scenarios of single-round and multi-round communication and validate it through numerical experiments. A bonus of our method is it does not reveal any information about the server's computation other than the information present in the output.

1 Introduction

Over the past decade, quantum computing technologies underwent a rapid series of advances. Both the size and power of quantum computers have been steadily increasing over the years, recently entering a new regime of "quantum supremacy"\cite{GoogleSupremacy}, in which the output of the quantum computations can be barely reproduced by the world's best supercomputers. Milestone achievements are the "quantum supremacy" demonstrations by Google\cite{Google2019} and USTC\cite{USTCSupremacy}, using superconducting qubits and photonic qubits, respectively.

A promising direction in quantum computing is the study of cloud computing scenarios, where a client requests a remote server to perform some desired quantum operations\cite{Arrighi2013, Fong2016, vanEnk2019}. However, limits on the amount of quantum communication between client and server constrain the size of the computations that can be effectively implemented in quantum cloud computing.

In this paper, we address the problem of reducing the amount of communication needed by the server to perform a quantum operation on quantum data provided by the client. The operation belongs to a parametric family of quantum gates known to both parties, but the specifications of the gate are known only to the server. Our objective here is to maximize the accuracy of the executed quantum operation when the capacity of the communication link and the total number of qubits exchanged between the server and the client are limited.

Our main contribution is a method for compressing a parametric family of quantum gates,
turning it into another gate family acting on a smaller number of qubits. Our method is based on autoencoders, a type of neural networks that have been very successful in classical machine learning [28], and have been recently used for the compression of quantum states [27]. We introduce a quantum gate autoencoder (QAEGate), providing the first quantum machine learning model that takes gates, rather than states, as inputs.

Our method also addresses the problem of minimizing the amount of information revealed to the client. In many situations, the server would not like to disclose any more information about the operation other than the unavoidable information revealed by the application of the operation on the client’s input. Our method achieves this feature by constructing a blind compression protocol, independent of the specifications of the operation.

Technically, the training of our QAEGate model is based on stochastic gradient descent [10]. We prove that the training is guaranteed to converge in polynomial time in the size of the initial operation. We then conduct numerical experiments that show the effectiveness of QAEGate in various settings including single-round and multi-round communication between the client and the server.

The remainder of the paper is organized as follows. In Section 2, we introduce some related works about quantum cloud computing and quantum autoencoders. Some preliminaries on autoencoders and quantum supermaps are provided in Section 3. Section 4 is devoted to introduce the quantum cloud computing task discussed in this paper. The structure and training details of our proposed QAEGate model are covered in Section 5. Then, we explain how to apply our method to concrete scenarios of quantum cloud computing in Section 6 and conduct some numerical experiments in Section 7. We conclude this paper in Section 8, with discussions on future directions for our method.

2 Related work

Classical cloud computing involves the execution of a computation on a remote server [3]. In the quantum version of this scenario [5], a client provides an input state and asks a remote server to perform a sequence of quantum gates on it. Two important issues are (1) to cope with the limited amount of quantum communication available in realistic scenarios, and (2) to minimize the leakage of the information about the server’s computation. These two issues have been previously approached with techniques from quantum Shannon theory [32] and quantum cryptography [30]. Related issues have also been considered in blind delegated quantum computation [11], where the goal is to guarantee data confidentiality on the client side rather than the server side.

Quantum autoencoders for quantum states were proposed by [27] and has been applied to quantum state compression and denoising quantum data [1, 9]. The quantum model used in these works has a similar structure to classical autoencoders, taking as input a quantum state represented by a fixed-length vector. This model, however, cannot be extended from quantum states to quantum gates, which in general are provided as black boxes. To convert a gate into a quantum state, one would have to apply it on a fixed input state. However, such conversion is not reversible due to the quantum No-programming theorem [25, 33].

3 Preliminaries

Autoencoders. An autoencoder is composed of an encoder and a decoder. One of its main application is dimensionality reduction, where the encoder maps a high-dimensional vector input $x$ to a low-dimensional representation $h$, and the decoder maps $h$ back to a reconstructed high-dimensional vector $x'$. In the training process, the autoencoder is optimized so that the high-dimensional vector $x'$ is as close as possible to the original high-dimensional vector $x$. If $x'$ and $x$ are close enough, the overall protocol provides a faithful compression of the original input $x$ into the low dimensional vector $h$. The structure of a typical autoencoder is depicted in Figure 1.

Quantum supermaps. A general quantum operation is represented by a quantum channel $\mathcal{C}$, which is a map from density operators to density operators [24]. A quantum supermap [13, 14] $\tilde{S}$ maps a quantum channel $\mathcal{C}$ into a quantum channel $\mathcal{C}'$ as $\mathcal{C}' = \tilde{S}(\mathcal{C})$. Every quantum supermap can be represented by a quantum circuit in Fig-
the quantum communication is an expensive resource, and it is often limited in realistic applications, making it difficult to transmit a full $n$-qubit quantum state back and forth between the client and the server. Here we consider the scenario where the capacity of the quantum communication link is bounded by $a \leq n$ qubits.

In order to cope with the bottleneck on the amount of quantum communication, we design a pair of quantum circuits, including an encoder $E$ implemented by the server and a decoder $D$ implemented by the client. The encoder $E$ serves as a quantum supermap transforming an $n$-qubit quantum channel to an $a$-qubit quantum channel, while the decoder $D$ does the opposite, recovering the $n$-qubit quantum channel from the output of the encoder. Since the decoder is implemented by the client, who has no knowledge of the parameter $x$, the supermap $D$ must be independent of $x$. On the other hand, the encoder is implemented by the server, and could in principle depend on $x$. However, a dependence on $x$ may result into a leakage of information to the client. For this reason, we also require the encoder $E$ to be independent of $x$. We will later show that this choice reaches the best possible confidentiality in the ideal situation where $U_x$ is accurately implemented.

5 Quantum Autoencoders for Quantum Gates

In this section we consider a basic scenario of quantum cloud computing with one round of communication between client and server. In this scenario, we develop a model of quantum autoen-
coders for quantum gates (QAEGate). In the following, we will introduce the structure of the proposed model and demonstrate its implementation and training through numerical experiments. Furthermore, we prove that stochastic gradient descent has convergence guarantee in the training of our QAEGate model. We will extend our model to more scenarios in Section 6.

5.1 Structure

Our model consists of an encoder and a decoder. The encoder produces an encoded quantum gate by inserting the gate $U_x$ into a suitable quantum circuit, initializing $n - a$ input qubits to a fixed state $|0\rangle$, and discarding $n - a$ output qubits. The result is a (generally noisy) quantum channel acting on $a$ qubits. In turn, the decoder converts the $a$-qubit channel back into an $n$-qubit channel, which aims to approximate the initial gate. The structure of the encoder and the decoder are shown in Figure 4.

To construct the encoding and decoding circuits, we use parameterized unitary operators, which have been successfully employed to build variational quantum circuits [16, 18]. Our construction is depicted in Figure 4. The encoder consists of two parameterized unitary operators, denoted by $U^{\theta_{le}}$ and $U^{\theta_{re}}$, depending on parameters $\theta_{le} = (\theta_{le1}, \theta_{le2}, ..., \theta_{lem})$ and $\theta_{re} = (\theta_{re1}, \theta_{re2}, ..., \theta_{rem})$. The gates $U^{\theta_{le}}$ and $U^{\theta_{re}}$ are placed on the left and right of the original quantum gate, respectively. The decoder consists of two unitary operators $U^{\theta_{ld}}$ and $U^{\theta_{rd}}$, depending on parameters $\theta_{ld} = (\theta_{ld1}, \theta_{ld2}, ..., \theta_{ldm})$ and $\theta_{rd} = (\theta_{rd1}, \theta_{rd2}, ..., \theta_{rdm})$. The gates $U^{\theta_{ld}}$ and $U^{\theta_{rd}}$ are placed on the left and right of the encoded gate, combined with the identity gate on the remaining $n - a$ qubits. Note that these parameterized unitary operators are independent of the parameter $x$ of the original quantum gate here.

The choice of the parameters $\theta_{le}$, $\theta_{re}$, $\theta_{ld}$, and $\theta_{rd}$ is optimized in order to maximize the similarity between the decoded quantum gate and the original quantum gate. As a similarity measure, we use the overlap between the Choi operators [15, 19], due to the relative ease of evaluating this quantity in numerical experiments. As the optimization procedure, we will use a stochastic gradient descent method, described in Subsection 5.2.

Figure 4: Implementation details of QAEGate. The parameterized unitary operators in yellow constitute the encoder while the parameterized unitary operators in green constitute the decoder. In the encoding phase, we insert the gate $U_x$ in the middle of the encoder, initialize $n - a$ input qubits to a fixed state $|0\rangle$, and discard $n - a$ output qubits to obtain a quantum channel acting on $a$ qubits. In the decoding phase, the decoder maps the channel back to an approximation of the initial $n$-qubit quantum gate.

To run the optimization, one needs first to fix the parametrization of the gates. The naive choice would be to pick a parametrization that can represent arbitrary quantum gates. However, this approach has obvious difficulties:

- Describing an arbitrary $n$-qubit unitary transformation requires an exponential number of parameters, and therefore a full optimization of the parameters in the autoencoder is only feasible for small values of $n$.
- Due to hardware restrictions of the current quantum computers, it is difficult to implement arbitrary $n$-qubit unitary operators. This is especially true for the client, who may only have the ability to implement a small number of basic quantum gates.

To address these problems, we propose an implementation of QAEGate based on an approximation of generic $n$-qubit unitary transformations that uses a polynomial number of parameters. We implement the parameterized unitary operators for $n$ qubits by decomposing them into a sequence of parameterized two-qubit unitary operators, where each pair of qubits corresponds to one such operator. Figure 5 presents the case when $n = 4$. For each two-qubit parameterized unitary operator, it is further decomposed into a sequence of basic quantum gates, as depicted in Figure 6.

Here $R_X(\theta)$ is the quantum gate defined as fol-
Figure 5: Decomposition of a parameterized 4-qubit unitary gate into 6 parameterized two-qubit gates.

Figure 6: Decomposition of a two-qubit parameterized unitary gate.

The other two gates $R_Y(\theta)$ and $R_Z(\theta)$ are similar to $R_X(\theta)$ but for Pauli matrices $\sigma_y$ and $\sigma_z$.

The two-qubit gate $XX(\theta)$ is the Ising coupling gate that commonly appears in real-world quantum computers with trapped-ion architecture [17]. It is defined as $XX(\theta) := \exp(i \theta \sigma_x \otimes \sigma_x)$ for the Pauli matrix $\sigma_x$, and written in matrix form as:

$$XX(\theta) = \begin{bmatrix}
\cos \theta & 0 & 0 & -i \sin \theta \\
0 & \cos \theta & -i \sin \theta & 0 \\
0 & -i \sin \theta & \cos \theta & 0 \\
-i \sin \theta & 0 & 0 & \cos \theta
\end{bmatrix}$$

$YY(\theta)$ and $ZZ(\theta)$ are Ising coupling gates corresponding to Pauli matrices $\sigma_y$ and $\sigma_z$ respectively.

5.2 Training

To find the optimal parameters of QAEGate, we adopt classical gradient-based methods such as stochastic gradient descent [20].

We denote the parameters of QAEGate to be trained as $\theta$. The training set is composed of $k$ quantum gates sampled randomly from $\{U_x\}$. In every iteration, we select a quantum gate $U$ from these $k$ gates randomly and insert it into the QAEGate. Then we calculate the overlap $f$ between the Choi state $C$ of $U$ and the Choi state $C'$ of the decoded state from the QAEGate. We define the loss function as $L = 1 - f$ here and update the parameters of QAEGate according to the gradients $\nabla_\theta L$ until the decoded quantum gate is close enough to the original quantum gate or the maximum number of iterations is reached. We describe the complete training process in Algorithm 1.

Algorithm 1: Quantum autoencoders for quantum gates (QAEGate).

Data: $k$ quantum gates sampled randomly from $\{U_x\}$, maximum number of iterations $K$, learning rate $\eta$, threshold $\delta$.

Initialize QAEGate parameters $\theta$ randomly, $i = 0$, $f = 1$

while $i < K$ or $f > \delta$ do

Select the quantum gate $U$ randomly from $k$ input gates and insert it into the QAEGate

Calculate the Choi state $C$ of $U$

Get the decoded state from the QAEGate and calculate its corresponding Choi state $C'$

$f = \text{Tr}(CC')$, which is the overlap between $C$ and $C'$

$L = 1 - f$

Calculate $\nabla_\theta L$ and update $\theta$ as $\theta = \theta - \eta \nabla_\theta L$

$i = i + 1$

end while

5.3 Convergence Analysis

We further prove that our proposed QAEGate can be efficiently trained by the stochastic gradient-based method in Algorithm 1. Specifically, we obtain the following convergence guarantee:

Theorem 1 (Convergence guarantee). If we perform SGD, as in Algorithm 1, to optimize $\theta$ in the training of the QAEGate model, then the convergence rate is $T = O\left(\frac{\text{dim} \theta}{\epsilon^2}\right)$ for achieving the condition $\mathbb{E}[\|\nabla_\theta L\|^2] \leq \epsilon^2$.

The proof is provided in the Appendix.

6 Applications

In this section we first apply QAGate to a basic scenario of the quantum cloud computing, in which we just consider single-round communication between the client and the server. Also, there is just one family of $n$-qubit quantum gates...
We first consider how much information one can obtain observing a qubit channel implemented by the server into an arbitrary malicious client is no larger than the protocol, the information about the approximation of the target n-qubit channel implemented by the server into an end implement the decoder, transforming the overall, the operations performed at the client’s qubits previously stored in the quantum memory.

An advantage of the structure of QAEGate is that it can fulfill the server’s confidentiality. Specifically, the server wants to forbid the client from obtaining the detailed implementation of the gates, which is described by \( \theta \). The most general way for the client to extract information about \( \theta \) is to insert it into a quantum circuit and make measurements on the output of the circuit, as shown in Figure 7(a). Let the output of the circuit to be \( Y \), which is a random variable correlated with \( X \). We write \( Y = F(U_X) \) to denote the relationship between \( X \) and \( Y \). The information one can know about \( X \), by obtaining the value of \( Y \), is the mutual information \( I(X;Y) \).

Next, we consider a malicious client who wants to obtain the most information about \( x \). The server interacts with the client via the encoder, which is equivalent to that the client has access to the encoded gate, which is a channel \( E(U_x) \). The most general way for the client to extract information about \( x \) is to insert \( E(U_x) \) into a circuit and measure the output of the circuit, as shown in Figure 7(b). Let this circuit be \( D' \) and its output be \( Y' \), which is related to \( X \) by \( Y' = (D' \circ E)(U_X) \). The maximum information the client could obtain is \( \max_{D'} I(X;Y') \).

Now, since \( E \) is determined by the server, the circuit \( D' \circ E \) with variable \( D' \) forms a subset of the range of \( F \), which contains all possible cir-
cuits taking $U_x$ as input. Therefore, the optimization over $D'$ is upper bounded by the optimization over $F$, and thus we have

$$\max_{D'} I(X; Y') \leq I_{\text{max}}.$$  \hspace{1cm} (1)

The equation above proves Theorem 2, since the left hand side is the maximal information obtainable by an arbitrary malicious client, and the right hand side is the inevitable information leakage given one access to $U_x$.

6.2 Multi-round communication scenario

In this scenario, we relax the assumption of one-round communication and the client can communicate with the server for multiple rounds. In each round, the client can send to and receive from the server an $a$-qubit quantum state through the quantum communication link. For this multi-round communication scenario, the server can exploit $U_x$ more than once. Here we depict the case of two-round communication in Figure 8.

For the optimal performance, a general protocol may utilize different encoders and decoders for each round of communication. Here we keep the remaining $n - a$ qubits of the output of the encoder in the first round and regard it as a part of the input of the encoder in the second round. We depict such a variation of the structure of the QAEGate model in Figure 9. The encoders and decoders of the first and second round are separately parameterized.

6.3 Sequence of gates scenario

In this scenario, we consider a more general situation that the client expects the server to execute a sequence of gates from different parametric families. In Figure 10, we depict the simplest case that the length of the sequence is 2. Here there are two classes of quantum gates, $\{U_x\}_{x \in X}$ and $\{U_y\}_{y \in Y}$, stored in the server. The client wants to obtain $U_y(U_x(\rho))$ by communicating with the server.

In order to handle this scenario, we design the encoder and the decoder with different structures and depict the modified QAEGate model in Figure 11.

The encoder is composed of four parameterized unitary operators, denoted by $U^{\theta_{x1}}, U^{\theta_{x2}}, U^{\theta_{x3}}$ and $U^{\theta_{x4}}$. The first two operators are placed on both sides of $U_x$ while the other two are placed on both sides of $U_y$. The decoder is composed of three parameterized parameterized unitary operators, denoted by $U^{\theta_{d1}}, U^{\theta_{d2}}$ and $U^{\theta_{d3}}$. 

Figure 9: QAEGate for multi-round communication scenario.

Figure 8: The multi-round communication scenario. The client can communicate with the server for multiple rounds.

Figure 10: Sequence of gates scenario. The client expects the server to execute a sequence of gates from different parametric families.

Figure 11: QAEGate for sequence of gates scenario.
7 Numerical Experiments

In this section, we examine the performance of our proposed models by numerical simulation. We show the effectiveness of our proposed models in different settings for gates based on the Heisenberg model. All simulations are implemented by Tensorflow quantum [12] and performed on a single GPU. We train the models by stochastic gradient descent in all of the experiments.

7.1 Heisenberg model

Heisenberg model [6] is a famous statistical mechanical model, which describes a magnetic system of half spins. It can be defined by the Hamiltonian

$$\hat{H} = -\frac{1}{2} \sum_{j=1}^{n-1} (J_x \sigma^x_j \sigma^x_{j+1} + J_y \sigma^y_j \sigma^y_{j+1} + J_z \sigma^z_j \sigma^z_{j+1}) - \frac{1}{2} \sum_{j=1}^{n} h \sigma^z_j,$$

where $J_x$, $J_y$, $J_z$ and $h$ are constants representing the strength of the coupling and the external magnetic field, respectively. $\sigma^x$, $\sigma^y$, $\sigma^z$ are the Pauli matrices. $n$ is the number of qubits in the quantum system. We define Heisenberg gates as the quantum gates which represent evolution defined by a Hamiltonian operator of Heisenberg model. These Heisenberg gates can be represented by

$$U(t) = \exp(-i\hat{H}t),$$

where $\hat{H}$ is the Hamiltonian of a Heisenberg model and $t$ is the evolution time.

7.2 Simulation results

**Basic Scenario.** In this scenario, we set $\{U_x\}$ stored in the sever as a class of Heisenberg gates with $J_x = J_y = J_z = 0.1$ and $h = 0.5$.

Here we define the evolution time $t$ as the parameter $x$, unknown to the client. The client can only transmit $a = 1$ qubit or $a = 2$ qubits through the quantum communication link. We consider three cases, $n = 2, 3, 4$, in the experiment. We exploit a training set composed of 50 gates to optimize the compression model and validate it on the test set composed of other 10 gates in each experiment.

We show the training curves of the QAEGate model in Figure 12. Here the x-axis represents the epoch number of the training, and the y-axis represents the overlap between the Choi states of the original gate and the decoded gate. We can see that all of the training processes converge finally, which conforms to our convergence analysis in Theorem 1.

![Figure 12: Training curves of the QAEGate for the basic scenario.](image)

Furthermore, we present the performance of our QAEGate model on test sets in Figure 13. The y-axis represents the average overlap values between the Choi states of the decoded quantum gate and the original quantum gate. We can see that our QAEGate model achieves satisfactory performance in all of the experiments, which confirms the effectiveness of our proposed communication model empirically. However, we can observe that QAEGate performs better when the dimension of the class of Heisenberg gates $\{U_x\}$ is small.

![Figure 13: The performance of our QAEGate model on test sets of the basic scenario.](image)
Multi-round communication scenario. In this scenario, we allow two-round communication between the server and the client while they can only transmit $a = 1$ qubit through the quantum communication link at a time.

We present the performance of our QAEGate model on test sets of the two-round communication scenario in Figure 14. The results indicate that our proposed model performs better when additional round of communication is available, compared with the basic scenario with $a = 1$. Especially for the case of $n = 4$, the average overlap value increases by 0.12 compared with the basic single-round scenario. We also compare the case of $a = 2$ in the basic scenario and the case of $a = 1$ in the two-rounds communication scenario. In both cases, two qubits are sent to and received from the server in total. We find that our proposed models have similar performance.

Figure 14: The performance of our QAEGate model on test sets of the multi-round communication scenario.

Sequence of gates scenario. In this scenario, we set $\{U_x\}$ stored in the server as a class of Heisenberg gates with $J_x = J_y = J_z = 0.1$ and $h = 0.5$ and $\{U_y\}$ as a class of Heisenberg gates with $J_x = J_y = 0.1$ and $J_z = h = 0.5$. Here we define the evolution times of them as $x$ and $y$ whose values can be different. The client still can only transmit $a = 1$ qubit through the quantum communication link. We also consider three cases, $n = 2, 3, 4$, in the experiment. The simulation results on test sets of this scenario are shown in Figure 15 and our proposed model achieved satisfactory performance in all three cases.

Figure 15: The performance of our QAEGate model on test sets of the sequence of gates scenario.

8 Discussion and conclusions

We proposed a quantum machine learning model, QAEGate, for communication-efficient quantum cloud computing, and applied it to single-round and multi-round communication scenarios. In the single-round scenario, the protocol avoids leakage of information about the server’s computation and achieves a satisfactory performance in terms of overlap with the target gate versus the number of qubits in its compressed version. By adjusting the structure of QAEGate, we applied our method to other two scenarios, involving multiple gates and multiple rounds of communication between the client and server.

The performance of our method has been tested numerically in a number of examples. Due to the exponential complexity of simulating quantum gates on classical computers, such analysis is necessarily limited to scenarios involving a small number of qubits. On the other hand, our method is meant to be eventually run on real quantum computers, on which the complexity of QAEGate grows only polynomially. While the algorithm is currently in its early stages, our results suggest that a real-world test may take place not too far in the future.

An application of the techniques developed in this paper is learning unknown quantum gates [8, 21, 29]. In this task, the goal is to imprint an unknown gate into the state of a quantum memory from which the gate can be accessed at a later time. Our QAEGate can be adapted to this task by adjusting the structure of the encoder and the decoder so that quantum gates can be encoded into quantum states. Referring to Figure 4, the idea is to choose different number of qubits $a$ and $a'$ for the input and output wires of the encoder, respectively. In particular, one can set $a$ to zero, meaning that the encoder produces a quantum state of $a'$ qubits. The number of qubits $a'$ can be regarded as the size of the quantum memory of a
learning machine designed to learn a generic gate from a given parametric family of quantum gates. An interesting feature of our method is that it allows one to set the memory size (i.e., to fix $a'$) and to achieve approximate learning subject to constraints on the available quantum memory.

Another interesting direction of future research is assessing the power of quantum processors [7, 22]. It has been often pointed out that the number of qubits in a quantum processor is not an appropriate measure, because the presence of noise generally limits the set of achievable computations. Hence, it is important to have more inclusive measures, that take into account not only the raw number of qubits, but also the size of the set of operations implementable on them. The approach of quantum gate compression sheds light into this problem by providing a rigorous notion of “effective size” of a quantum processor, defined as the minimum number of qubits on which the operations of the processor can be compressed with sufficiently high fidelity.

More specifically, the set of operations implemented by a given quantum processor can be regarded as a parametric family of (generally noisy) quantum gates. When the processor is not universal, such a family may be compressible into operations acting on a smaller number of qubits, meaning that the computations achieved by the processor can be in principle simulated with a smaller quantum computer. The size of this minimal simulator can then be regarded as a measure of the computing power of the original quantum processor. Such a measure is independent of the hardware implementation, and could be in principle evaluated by connecting the given processor to a reference quantum computer that runs the QAEGate algorithm, or a generalization of it working with noisy input gates.

A limitation of the above approach is that the evaluation of the minimum number of qubits requires a trusted quantum computer of the same size of the original processor. On the other hand, once a sufficiently large universal quantum computer becomes available, it can be used as a standard reference for assessing the power of other quantum processors, and to construct efficient simulations that can be run on universal quantum computers of smaller size.

References

[1] Tom Achache, Lior Horesh, and John Smolin. Denoising quantum states with quantum autoencoders—theory and applications, 2020. arXiv:2012.14714
[2] Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. arXiv preprint arXiv:1708.08694, 2017. arXiv:1708.08694
[3] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. Communications of the ACM, 53(4):50–58, 2010.
[4] Frank Arute et al. Quantum supremacy using a programmable superconducting processor. Nature, 574(7779):505–510, 2019. arXiv:1910.11333
[5] Stefanie Barz, Elham Kashefi, Anne Broadbent, Joseph F Fitzsimons, Anton Zeilinger, and Philip Walther. Demonstration of blind quantum computing. science, 335(6066):303–308, 2012.
[6] Rodney J Baxter. Exactly solved models in statistical mechanics. Elsevier, 2016.
[7] Lev S Bishop, Sergey Bravyi, Andrew Cross, Jay M Gambetta, and John Smolin. Quantum volume. Quantum Volume. Technical Report, 2017.
[8] Alessandro Bisio, Giulio Chiribella, Giacomo Mauro D’Ariano, Stefano Facchini, and Paolo Perinotti. Optimal quantum learning of a unitary transformation. Physical Review A, 81(3):032324, 2010. arXiv:0903.0543
[9] Dmytro Bondarenko and Polina Feldmann. Quantum autoencoders to denoise quantum data. Physical Review Letters, 124(13):130502, 2020. arXiv:1910.09169
[10] Léon Bottou. Stochastic gradient descent tricks. In Neural networks: Tricks of the trade, pages 421–436. Springer, 2012.
[11] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Universal blind quantum computation. In 2009 50th Annual IEEE Symposium on Foundations of Computer Science, pages 517–526. IEEE, 2009.
[12] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J. Martinez, Jae Hyeon Yoo, Sergei V. Isakov, Philip
Massey, Murphy Yuezhen Niu, Ramin Halavati, Evan Peters, et al. Tensorflow quantum: A software framework for quantum machine learning, 2020. arXiv:2003.02989

[13] Giulio Chiribella, Giacomo Mauro D’Ariano, and Paolo Perinotti. Transforming quantum operations: Quantum supermaps. *EPL (Europhysics Letters)*, 83(3):30004, 2008.

[14] Giulio Chiribella, Giacomo Mauro D’Ariano, and Paolo Perinotti. Theoretical framework for quantum networks. *Physical Review A*, 80(2):022339, 2009. arXiv:0904.4483

[15] Man-Duen Choi. Completely positive linear maps on complex matrices. *Linear algebra and its applications*, 10(3):285–290, 1975.

[16] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019. arXiv:1810.03787

[17] Shantanu Debnath, Norbert M Linke, Caroline Figgatt, Kevin A Landsman, Kevin Wright, and Christopher Monroe. Demonstration of a small programmable quantum computer with atomic qubits. *Nature*, 536(7614):63–66, 2016. arXiv:1603.04512

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

[19] Andrzej Jamiołkowski. Linear transformations which preserve trace and positive semidefiniteness of operators. *Reports on Mathematical Physics*, 3(4):275–278, 1972.

[20] Jack Kiefer, Jacob Wolfowitz, et al. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[21] Yin Mo and Giulio Chiribella. Quantum-enhanced learning of rotations about an unknown direction. *New Journal of Physics*, 21(11):113003, 2019. arXiv:1906.01300

[22] Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, Daniel J Egger, Stefan Filipp, Andreas Fuhrer, Jay M Gambetta, Marc Ganzhorn, et al. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology*, 3(3):030503, 2018. arXiv:1710.01022

[23] Tomoyuki Morimae and Keisuke Fujii. Blind quantum computation protocol in which alice only makes measurements. *Physical Review A*, 87(5):050301, 2013.

[24] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[25] Michael A Nielsen and Isaac L Chuang. Programmable quantum gate arrays. *Physical Review Letters*, 79(2):321, 1997. arXiv:9703032

[26] John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, 2018. arXiv:1801.00862

[27] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, 2017. arXiv:1612.02806

[28] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. arXiv:1404.7828

[29] Michal Sedlák and Mário Ziman. Probabilistic storage and retrieval of qubit phase gates. *Physical Review A*, 102(3):032618, 2020. arXiv:2008.09555

[30] Yu-Bo Sheng and Lan Zhou. Distributed secure quantum machine learning. *Science Bulletin*, 62(14):1025–1029, 2017.

[31] Ryan Sweke, Frederik Wilde, Johannes Jakob Meyer, Maria Schulz, Paul K Fährmann, Barthélémy Meynard-Piganeau, and Jens Eisert. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum*, 4:314, 2020.

[32] Yuxiang Yang, Giulio Chiribella, and Masahito Hayashi. Communication cost of quantum processes. *IEEE Journal on Selected Areas in Information Theory*, 1(2):387–400, 2020. arXiv:2002.06840

[33] Yuxiang Yang, Renato Renner, and Giulio Chiribella. Optimal universal programming of unitary gates. *Physical Review Letters*, 125(21):210501, 2020. arXiv:9703032

[34] Han-Sen Zhong et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020. arXiv:2012.01625
A Proof of Theorem 1

According to [2], we have the following lemma about convergence of SGD for nonconvex functions:

**Lemma 1.** If a function $f(x)$ is $L_1$-smooth and we perform SGD update $x_{t+1} \leftarrow x_t - \eta \nabla f_i(x_t)$ each time for a random $i \in [n]$, then the convergence rate is $T = \mathcal{O} \left( \frac{L}{\eta^2} \right)$ for finding $\mathbb{E} [\| \nabla f(x) \|^2] \leq \epsilon^2$.

Therefore, we first discuss the smoothness property of the loss function $\mathcal{L}(\theta)$ in the training of QAEGate model. Note that the first- and second-order Lipschitz smoothness are equivalent to Lipschitz continuity of the first- and second-order derivatives. So we start with a lemma about the Lipschitz continuity of multivariate functions [31].

**Lemma 2.** Given some function $f : \mathbb{R}^M \rightarrow \mathbb{R}$, if all partial derivatives of $f$ are continuous, then for any $a, b \in \mathbb{R}$ the function $f : [a, b]^M \rightarrow \mathbb{R}$ is $L$-Lipschitz continuous with

$$
L = \sqrt{M} \max_{j \in \{1, \ldots, M\}} \sup_{x \in [a, b]^M} \left| \frac{\partial f(x)}{\partial x_j} \right|.
$$

Next we prove a more general result about the smoothness of measurement probability of quantum circuits, and then show that $\mathcal{L}(\theta)$ fits into this result.

**Lemma 3.** Consider a quantum circuit consisting of any number of fixed unitary gates and $M$ variable unitary gates $U_1(\theta_1), \ldots, U_M(\theta_M)$, where $U_j(\theta_j) := \exp(i \theta_j H_j)$ for some Hermitian operator $H_j$. Then the probability of any measurement outcome of the output of this circuit is $L_1$-smooth and $L_2$-second-order smooth with respect to $\theta = (\theta_1, \ldots, \theta_M)$, where

$$
L_1 = 4MH_\text{max}^2
$$

$$
L_2 = 8M^{3/2}H_\text{max}^3
$$

where $H_\text{max} := \max_j \|H_j\|_2$ and $\| \cdot \|_2$ is the induced operator norm defined as $\|A\|_2 := \sup \frac{\|Ax\|_2}{\|x\|_2}$.

**Proof.** Let $|\psi_0\rangle$ be the initial state of the circuit. Without loss of generality, we assume the variable unitary gates are labeled as $U_1(\theta_1)$ to $U_M(\theta_M)$ according to the order they are applied. Then the output of this circuit can be written as $|\psi_\theta\rangle := V_M U_M(\theta_M) \cdots V_1 U_1(\theta_1) V_0 |\psi_0\rangle$, where $V_1, \ldots, V_M$ denotes the fixed unitary gates. A measurement outcome can be described by a positive operator-valued measure (POVM) element $P$ with the property that $P$ and $I - P$ are both positive semidefinite, where $I$ is the identity operator of the output. The probability of this outcome is

$$
f(\theta) := \text{Tr}[P |\psi_\theta\rangle \langle \psi_\theta|] = \langle \psi_\theta | P |\psi_\theta\rangle.
$$

To study the smoothness of $f(\theta)$, we take the partial derivatives:

$$
\frac{\partial f(\theta)}{\partial \theta_j} = \frac{\partial \langle \psi_\theta | P |\psi_\theta\rangle}{\partial \theta_j} + \langle \psi_\theta | P \frac{\partial |\psi_\theta\rangle}{\partial \theta_j} \rangle = 2 \Re \left( \frac{\partial \langle \psi_\theta | P |\psi_\theta\rangle}{\partial \theta_j} \right)
$$

$$
\frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} = 2 \Re \left( \frac{\partial^2 \langle \psi_\theta | P |\psi_\theta\rangle}{\partial \theta_j \partial \theta_k} + \frac{\partial \langle \psi_\theta | P \frac{\partial |\psi_\theta\rangle}{\partial \theta_j} \rangle}{\partial \theta_k} + \frac{\partial \langle \psi_\theta | P \frac{\partial |\psi_\theta\rangle}{\partial \theta_k} \rangle}{\partial \theta_j} \right)
$$

$$
\frac{\partial^3 f(\theta)}{\partial \theta_j \partial \theta_k \partial \theta_l} = 2 \Re \left( \frac{\partial^3 \langle \psi_\theta | P |\psi_\theta\rangle}{\partial \theta_j \partial \theta_k \partial \theta_l} + \frac{\partial^2 \langle \psi_\theta | P \frac{\partial |\psi_\theta\rangle}{\partial \theta_j} \rangle}{\partial \theta_k \partial \theta_l} + \frac{\partial^2 \langle \psi_\theta | P \frac{\partial |\psi_\theta\rangle}{\partial \theta_k} \rangle}{\partial \theta_j \partial \theta_l} + \frac{\partial \langle \psi_\theta | P \frac{\partial^2 |\psi_\theta\rangle}{\partial \theta_j \partial \theta_k} \rangle}{\partial \theta_l} + \frac{\partial \langle \psi_\theta | P \frac{\partial^2 |\psi_\theta\rangle}{\partial \theta_j \partial \theta_l} \rangle}{\partial \theta_k} + \frac{\partial \langle \psi_\theta | P \frac{\partial^2 |\psi_\theta\rangle}{\partial \theta_k \partial \theta_l} \rangle}{\partial \theta_j} \right)
$$

These partial derivatives can be bounded with vector and operator norms. For a unitary operator $U$, $\|U\|_2 = 1$. For the POVM element $P$, $\|P\|_2 \leq 1$. The second-order partial derivatives can be
bounded as:
\[
\left| \frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} \right| \leq 2 \left( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_j \partial \theta_k} \right| P|\psi_\theta| \right) + \left| \frac{\partial (\psi_\theta)}{\partial \theta_j} P \frac{\partial (\psi_\theta)}{\partial \theta_k} \right| \\
\leq 2 \left( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_j \partial \theta_k} \right| P \|P\|_2 \|\psi_\theta\|_2 \right) + 2 \left( \left| \frac{\partial (\psi_\theta)}{\partial \theta_j} \right| P \|P\|_2 \left| \frac{\partial (\psi_\theta)}{\partial \theta_k} \right| \right) \\
\leq 2 \max_{a,b} \left( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_a \partial \theta_b} \right| \right) + 2 \max_{a} \left( \left| \frac{\partial (\psi_\theta)}{\partial \theta_a} \right| \right)^2
\]
According to Eqs. (6) and (7), we obtain
\[
\begin{align*}
\frac{\partial^3 f(\theta)}{\partial \theta_j \partial \theta_k \partial \theta_l} &\leq 2 \left| \frac{\partial^3 \psi_\theta}{\partial \theta_j \partial \theta_k \partial \theta_l} \right| P|\psi_\theta| + \left| \frac{\partial^2 \psi_\theta}{\partial \theta_j \partial \theta_k} \right| P \left( \frac{\partial (\psi_\theta)}{\partial \theta_l} \right) + \left| \frac{\partial^2 \psi_\theta}{\partial \theta_j \partial \theta_l} \right| P \left( \frac{\partial (\psi_\theta)}{\partial \theta_k} \right) + \left| \frac{\partial \psi_\theta}{\partial \theta_j} \right| P \left( \frac{\partial^2 \psi_\theta}{\partial \theta_k \partial \theta_l} \right) \\
&\leq 2 \max_{a,b,c} \left( \left| \frac{\partial^3 \psi_\theta}{\partial \theta_a \partial \theta_b \partial \theta_c} \right| \right) + 6 \left( \max_{a,b} \left( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_a \partial \theta_b} \right| \right) \right) \left( \max_{a} \left( \left| \frac{\partial \psi_\theta}{\partial \theta_a} \right| \right) \right)^2
\end{align*}
\]
Therefore, to show that the second- and third-order partial derivatives are bounded, it suffices to show that max \( \left| \frac{\partial (\psi_\theta)}{\partial \theta_a} \right| \right|_{a,b} \) and third-order partial derivatives are bounded, it suffices to show that max \( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_a \partial \theta_b} \right| \right|_{a,b,c} \) are all bounded. We observe that \( \frac{\partial \psi_\theta}{\partial f} = \frac{\partial \exp(\theta, H_f)}{\partial f} = iH_f U_j(\theta_j) \), and we have
\[
\max_{a} \left| \frac{\partial (\psi_\theta)}{\partial \theta_a} \right| \right|_{a,b} = \max_{a} \|V_M U_M(\theta_M) \ldots V_a iH_f U_a(\theta_a) \ldots U_1(\theta_1)V_0|\psi_\theta\|_2 \\
\leq \max_{a} \|V_M U_M(\theta_M) \ldots V_a \|_2 \|H_f \|_2 \|U_a(\theta_a) \ldots U_1(\theta_1)V_0|\psi_\theta\|_2 \\
= \max_{a} \|H_f \|_2 = H_{\max}
\]
With similar derivations, we can obtain max \( \left| \frac{\partial \psi_\theta}{\partial \theta_a} \right| \right|_{a,b,c} \) and third-order partial derivatives are bounded, it suffices to show that max \( \left| \frac{\partial^2 \psi_\theta}{\partial \theta_a \partial \theta_b} \right| \right|_{a,b,c} \) are all bounded. We observe that \( \frac{\partial \psi_\theta}{\partial f} = \frac{\partial \exp(\theta, H_f)}{\partial f} = iH_f U_j(\theta_j) \), and we have
\[
\max_{a} \left| \frac{\partial (\psi_\theta)}{\partial \theta_a} \right| \right|_{a,b} = \max_{a} \|V_M U_M(\theta_M) \ldots V_a iH_f U_a(\theta_a) \ldots U_1(\theta_1)V_0|\psi_\theta\|_2 \\
\leq \max_{a} \|V_M U_M(\theta_M) \ldots V_a \|_2 \|H_f \|_2 \|U_a(\theta_a) \ldots U_1(\theta_1)V_0|\psi_\theta\|_2 \\
= \max_{a} \|H_f \|_2 = H_{\max}
\]
According to Eqs. (6) and (7), we obtain
\[
\left| \frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} \right| \leq 4H_{\max}^2 \\
\left| \frac{\partial^3 f(\theta)}{\partial \theta_j \partial \theta_k \partial \theta_l} \right| \leq 8H_{\max}^3
\]
Consider \( \frac{\partial f(\theta)}{\partial \theta_j} \) as a function of \( \theta \). Since its partial derivatives \( \frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} \) are bounded by \( 4H_{\max}^2 \), according to Lemma 2, \( \frac{\partial f(\theta)}{\partial \theta_j} \) is \( L'_1 \)-continuous with \( L'_1 = 4\sqrt{M}H_{\max}^2 \). Then
\[
\| \nabla f(\alpha) - \nabla f(\beta) \|_2 \leq \sqrt{M} \max_{j} \left| \frac{\partial f(\alpha)}{\partial \theta_j} - \frac{\partial f(\beta)}{\partial \theta_j} \right| \\
\leq \sqrt{M} L'_1\|\alpha - \beta\|_2 = 4M^2H_{\max}^2\|\alpha - \beta\|_2
\]
and therefore \( f \) is \( L_1 \)-smooth with \( L_1 = 4M^2H_{\max}^2 \). Similarly, consider \( \frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} \) as a function of \( \theta \). Since its partial derivatives \( \frac{\partial^3 f(\theta)}{\partial \theta_j \partial \theta_k \partial \theta_l} \) are bounded by \( 8H_{\max}^3 \), according to Lemma 2, \( \frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k} \) is \( L'_2 \)-continuous with \( L'_2 = 8\sqrt{M}H_{\max}^3 \). Then
\[
\| \nabla^2 f(\alpha) - \nabla^2 f(\beta) \|_2 \leq M \max_{j,k} \left| \frac{\partial^2 f(\alpha)}{\partial \theta_j \partial \theta_k} - \frac{\partial^2 f(\beta)}{\partial \theta_j \partial \theta_k} \right| \\
\leq ML'_2\|\alpha - \beta\|_2 = 8M^{3/2}H_{\max}^3\|\alpha - \beta\|_2
\]
and therefore \( f \) is \( L_2 \)-second-order smooth with \( L_2 = 8M^{3/2}H_{\max}^3 \). □
Based on above lemma, we obtain the following theorem:

**Theorem 3 (Smoothness).** The loss function $L(\theta)$ in the training of QAEGate model is $L_1$-smooth and $L_2$-second-order smooth.

**Proof.** Let $p(\theta)$ be the probability of outcome $|+\rangle$ in the SWAP test when the gates are parameterized with $\theta = (\theta_{le}, \theta_{re}, \theta_{ld}, \theta_{rd})$. Because of $L(\theta) = 1 - f(\theta)$ and $f(\theta) = 2p(\theta) - 1$, we just need to discuss the smoothness of $p(\theta)$. Since $p(\theta)$ is the probability of a measurement outcome of a unitary circuit, and in this circuit, every variable gate has the form of $U_\theta = \exp(i\theta H)$, we can apply Lemma 3. Thus, $p(\theta)$ is $L_1$-smooth and $L_2$-second-order smooth where $L_1 = 4M H_{\text{max}}^2$, $L_2 = 8M^{3/2} H_{\text{max}}^3$ and $M$ is the number of variable gates in the circuit. To prove Theorem 3, we only need to show that $H_{\text{max}}$ is bounded.

The variable gates in our circuit only includes the following single- and two-qubit gates:

$$R_X(\theta) := \exp(i\theta \sigma_x), \quad R_Y(\theta) := \exp(i\theta \sigma_y), \quad R_Z(\theta) := \exp(i\theta \sigma_z)$$

$$X X(\theta) := \exp(i\theta \sigma_x \otimes \sigma_x), \quad Y Y(\theta) := \exp(i\theta \sigma_y \otimes \sigma_y), \quad Z Z(\theta) := \exp(i\theta \sigma_z \otimes \sigma_z)$$

Therefore, $H_{\text{max}} = \max\{\|\sigma_x\|_2, \|\sigma_y\|_2, \|\sigma_z\|_2, \|\sigma_x \otimes \sigma_x\|_2, \|\sigma_y \otimes \sigma_y\|_2, \|\sigma_z \otimes \sigma_z\|_2\} = 1$. \qed

**Proof.** Based on Theorem 3, we have proved that $L(\theta)$ is $L_1$-smooth, where $L_1 = 4M$. In our implementation, there is only one variable for each variable unitary gate, so $M = \dim \theta$ here. Then we can apply Lemma 1 and obtain that the convergence rate is $T = O(\frac{1}{\epsilon^4})$ for finding $\mathbb{E}[\|\nabla_\theta L(\theta)\|^2] \leq \epsilon^2$. \qed