ICCAD Special Session Paper: Quantum Variational Methods for Quantum Applications

Shouvanik Chakrabarti
Department of Computer Science
University of Maryland
College Park, MD, USA
shouv@cs.umd.edu

Xuchen You
Department of Computer Science
University of Maryland
College Park, MD, USA
xyou@umd.edu

Xiaodi Wu
Department of Computer Science
University of Maryland
College Park, MD, USA
xwu@cs.umd.edu

***Abstract*—Quantum Variational Methods are promising near-term applications of quantum machines, not only because of their potential advantages in solving certain computational tasks and understanding quantum physics but also because of their feasibility on near-term quantum machines. However, many challenges remain in order to unleash the full potential of quantum variational methods, especially in the design of efficient training methods for each domain-specific quantum variational ansatzes. This paper proposes a theory-guided principle in order to tackle the training issue of quantum variational methods and highlights some successful examples.**

***Index Terms*—quantum variational methods, training, quantum generative adversarial networks, differentiable programming languages, auto-differentiation**

I. Introduction

With the rapid development of prototypes of quantum machines, especially the recent establishment of quantum supremacy [1], [2], the research of quantum computing has entered a new stage where near-term Noisy Intermediate-Scale Quantum (NISQ) computers [3] become the important platform for demonstrating quantum applications.

Quantum Variational Methods (QVMs) (e.g., [4]–[6]), or the so-called quantum neural-networks, gradually become a major candidate of quantum applications on NISQ machines. One appealing factor of adopting QVMs based on classically parameterized quantum circuits is their feasibility on near-term quantum machines. Indeed, such classically parameterized quantum circuits are predicted to be implementable at the scale of 100$\sim$200 qubits and a reasonable number of gates. Even though this scale of quantum circuits will not enable the implementation of famous quantum algorithms (e.g., Shor's and Grover's), they are already intractable to simulate by classical means. As a result, QVMs will enable efficient "quantum" mappings from input to output in the near term, which are already impossible to achieve with any classical means. How to leverage these quantum mappings is still an active research field. However, it is commonly believed that QVMs will help resolve quantum physics related computational problems in the near future. They are also likely helpful for solving general information/computational tasks, especially when the nature of these tasks exhibits certain structures that can be exploited by quantum mechanics. Finally, hinted by the significant success of deep learning, a potential paradigm change in application design might be emerging, where any new application/program could be obtained by training parameterized neural-networks (or programs) over data sets, rather than by a careful case-by-case design. In this sense, QVMs might serve as an important approach to achieve quantum applications even when fully-fledged quantum machines are developed.

This work received support from the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Accelerated Research in Quantum Computing and Quantum Algorithms Team programs, as well as the U.S.National Science Foundation grant CCF-1755800, CCF-1816695, and CCF-1942837 (CAREER).

Therefore, a lot of study has already been devoted to the design, analysis, and small-scale implementation of QVMs (e.g., see the survey [7]). One prominent example is the *variational quantum eigensolver*(VQE) [4] which is a QVM that finds the ground state/energy of physically-interesting Hamiltonian systems and finds promising applications in quantum chemistry. Another one is the *quantum approximate optimization algorithm* (QAOA) [5] which proposes a near-term feasible variational circuits that mimic the behavior of quantum adiabatic algorithms in solving optimization problems. One can also further leverage variational quantum circuits for classification [6], generative models [8], and several other learning tasks.

A lot of existing works on quantum variational methods have been focusing on the design of new applications that might benefit from the use of variational quantum circuits. However, an equally important topic, which concerns efficient training of these variational models, demonstrates a lot of challenges in unleashing the full potential of quantum variational methods. Indeed, the training of variational quantum circuits is known to be hard empirically, even on small instances of quantum circuits (e.g., $\geq$ 10 qubits), with some limited theoretical understanding (e.g., [9]).

In contrast to the study in classical machine learning, where empirical research has played a major role in investigating the training methods, empirical research in quantum variational methods is arguably restricted. This is due to the limitation of currently available quantum machines as well as the exponential complexity of simulating them by classical means. As a result, our current empirical findings will not necessarily generalize to intermediate-size variational quantum circuits, which are predicted to be available in the near future.

We propose a theory-guided principle in the research of QVM-related training issues with the hope to investigate the behavior of QVM instances beyond the current capability of empirical research. This approach becomes possible thanks to the recent development of theoretical understanding of classical machine learning, especially in the understanding of the landscape of the loss functions and regularization in training.

We will present the relevant preliminaries on quantum information, quantum neural networks and their comparison with classical neural networks in Section II. After that, we will illustrate a few successful attempts of ours following this theory-guided principle. Specifically,

In Section III, we demonstrate a quantitative investigation on the landscape of loss functions of quantum neural networks in [10] inspired by corresponding classical literature. Specifically, for typical under-parameterized quantum neural networks, there exists a dataset as shown in [10] that induces a loss function with the number of spurious local minima depending exponentially on the number of parameters. While local minima in classical neural networks are due to non-linear activations, in quantum neural networks local minima appear as a result of the quantum interference phenomenon.

In Section IV, we illustrate an example in the context of quantum generative adversarial networks (GANs), where the robustness and the scalability of training could be significantly improved by leveraging the Wasserstein semi-metric between quantum data [11]. This improvement also enables a surprising application where the proposed quantum Wasserstein GAN has been used to generate a 3-qubit quantum circuit of 50 gates that well approximates a 3-qubit 1-d Hamiltonian simulation circuit that requires over 10k gates using standard techniques.

Finally, in Section V, we tackle another important aspect of the training, which is the efficient calculation of gradients in quantum variational methods by quantum means. In particular, in [12], we develop an automatic differentiation technique not only for the plain variational quantum circuits but also for ones with program features (i.e., controls). Namely, we formalize the notion of *differentiable quantum programming languages*, inspired both by the success of differentiable programming languages in classical machine learning and the need of scalable gradient calculations for quantum variational methods. Furthermore, we showcase how differentiable quantum programming languages help with the training of more general variational quantum ansatzes that demonstrate more expressive power and hence save in the required quantum resources.

II. Preliminaries

Quantum states. Atomic building blocks of quantum computers are *qubits*. A qubit is a generalization of a classical binary bit that can be represented as a positive semidefinite trace-1 Hermitian (*density matrices*) in $\mathbb{C}^{2\times 2}$. The density matrix of a n-qubit state lies in the tensor product of spaces for single qubits $\otimes_{i=1}^{n}\mathbb{C}^{2\times 2}$, and is a linear operator with dimension $d = 2^n$. For quantum states on a Hilbert space

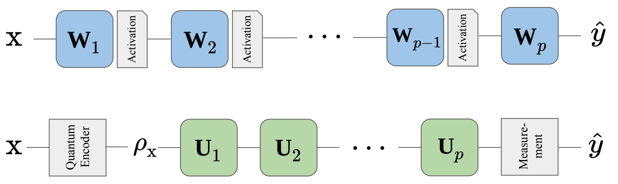

Fig. 1. Illustrations of classical and quantum neural networks. Both ClaNNs and QNNs have a layer structure, with distinctions in terms of the encoding, activation, measurement and parameterization.

$\mathbb{C}^{\mathcal{X}}$, let $\mathcal{D}(\mathcal{X})$ denote the set of all density matrices and $\mathcal{H}(\mathcal{X})$ the set of Hermitian matrices. A quantum state is said to be *pure* its density matrix is rank-1. A pure state can be represented by the vector $\mathbf{v} \in \mathbb{C}^{\mathcal{X}}$ such that $\boldsymbol{\rho} = \mathbf{v}\mathbf{v}^\dagger$. The dynamics of pure quantum states are represented by unitary matrices U such that $UU^\dagger = U^\dagger U = I$. The action $U(\boldsymbol{\rho})$ is given by $U\rho U^\dagger$. General quantum operations are described by ensembles $\{p_i, U_i\}$ of unitary matrices that maps a state $\boldsymbol{\rho}$ to $\sum_i p_i U_i \boldsymbol{\rho} U_i^\dagger$.

Quantum measurements and observables. A quantum measurement is specified by a set of matrices $\{\mathbf{E}_m\}$ such that $\sum_m \mathbf{E}_m^\dagger \mathbf{E}_m$ is identity. Given a quantum state $\boldsymbol{\rho}$, the outcome of the measurement is m with probability $\operatorname{tr}(\boldsymbol{\rho}\mathbf{E}_m^\dagger \mathbf{E}_m)$. Associating each outcome m with a real-valued scalar λ_m, the Hermitian $\mathbf{M} := \sum_m \lambda_m \mathbf{E}_m^\dagger \mathbf{E}_m$ is sometimes refer to as an observable. The expectation of an observable $\mathbf{M}$ on state $\boldsymbol{\rho}$ is denoted by $\mathcal{E}_M[\boldsymbol{\rho}] := \operatorname{tr}(\mathbf{M}\boldsymbol{\rho})$.

Quantum neural networks. Quantum neural networks (QNNs) are parameterized machine learning models based on variational quantum circuits. Similar to classical neural networks (ClaNNs), QNNs has a layered structure (Figure 1): for a p-layer QNN with input state $\boldsymbol{\rho}^{(0)}$, the intermediate states $\boldsymbol{\rho}^{(l)} = \mathbf{U}_l \boldsymbol{\rho}^{(l-1)} \mathbf{U}_l^\dagger$ for $l \in [p]$, where $\mathbf{U}_l$ is the matrix representation of the l-th layer, and $\boldsymbol{\rho}^{(p)}$ is the output state of the QNN. If the input state is pure (i.e. $\boldsymbol{\rho}^{(0)} = \mathbf{v}^{(0)}(\mathbf{v}^{(0)})^\dagger$ for some $\mathbf{v}^{(0)}$), the QNN can be equivalently expressed in the form of state vectors as $\mathbf{v}^{(l)} = \mathbf{U}_l \mathbf{v}^{(l-1)}$ for $l \in [p]$.

Comparison with Classical neural networks (ClaNNs). For comparison, typical feed-forward neural networks with scalar output are parameterized by a sequence of matrices $\{\mathbf{W}_l\}_{l=1}^{p}$, such that $\mathbf{W}_l \in \mathbb{R}^{d_l \times d_{l-1}}$ with $d_p = 1$. Given the input $\mathbf{x}^{(0)}$, the output $\hat{y}$ of the neural network is defined as

$$\hat{y} = \mathbf{W}_p \mathbf{x}^{(p-1)}, \text{ with } \mathbf{x}^{(l)} = \sigma(\mathbf{W}_{l-1}\mathbf{x}^{(l-1)}) \text{ for } l \in [p] \quad (1)$$

Here $\sigma(\cdot)$ denotes the (non-linear) activation on the output of each layer. Common choices for the activation function include non-linear element-wise mappings such as ReLU or Sigmoid.

Linear neural networks [13] is a special case of ClaNN where σ is chosen to be the identity mapping $\sigma(w) = w$. They can be represented as $\hat{y} = \mathbf{W}_p \mathbf{W}_{p-1} \cdots \mathbf{W}_1 \mathbf{x}^{(0)}$, which is similar to the vector representation of QNNs with pure input states (See Figure 1), but with differences in terms of input, parameterization and output scheme as specified below.

Input. The input to QNNs are quantum states. Therefore for a classical dataset, the feature vectors first need to be encoded into quantum states (common encoding scheme can be found in e.g., [7], [14], [15]). For generality, we don't make assumptions on the encoding scheme and will work directly with training sets composed of pairs of density matrices and labels $\mathcal{S} = \{(\boldsymbol{\rho}_i, y_i)\}_{i=1}^m$.

Parameterization. While the matrices $\{\mathbf{W}_l\}_{l=1}^p$ in ClaNNs could be any general matrices, the matrix representations $\{\mathbf{U}_l\}_{l=1}^p$ in QNNs must be proper quantum channels. Here we only consider unitary $\mathbf{U}_l$'s. Furthermore, in contrast to the directly parameterized matrices W_l, QNNs are typically composed of quantum gates parameterized by real parameter θ via matrix exponentials with the form $\exp(-i\theta\mathbf{H})$. The Hermitian $\mathbf{H}$ is referred to as the Hamiltonian that generates the rotation. Common choices of parameterized gates are Pauli rotations (e.g., [6], [16], [17]), generated by single-/multi-qubit Pauli matrices (i.e. tensor products of $\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\mathbf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$, $\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$).

While a layer in QNNs may consists of more than one parameterized rotation by grouping gates that can be executed in parallel, we assume one parameterized gate per layer without loss of generality:

$$\mathbf{U}(\boldsymbol{\theta}) = \mathbf{V}_p(\theta_p)\mathbf{V}_{p-1}(\theta_{p-1})\cdots\mathbf{V}_1(\theta_1), \tag{2}$$

where $\mathbf{V}_l(\theta_l) = e^{-i\theta_l\mathbf{H}_l}$ for Hamiltonian $\mathbf{H}_l$ and $l \in [p]$.

Output. The output of QNNs are real-valued scalars, so quantum *measurements* are performed on the output states to extract information from QNNs. The prediction $\hat{y}$ associated with an observable M on the output state $\boldsymbol{\rho}^{(p)}$ is given by $\hat{y} := \mathcal{E}_M[\boldsymbol{\rho}^{(p)}] = \operatorname{tr}(\boldsymbol{\rho}^{(p)}\mathbf{M})$. For a QNN with the matrix representation $\mathbf{U}(\boldsymbol{\theta})$, we use $f(\boldsymbol{\rho}, \boldsymbol{\theta})$ to denote the output with input state $\boldsymbol{\rho}$ and parameter $\boldsymbol{\theta}$: $f(\boldsymbol{\rho}, \boldsymbol{\theta}) := \operatorname{tr}(\mathbf{U}(\boldsymbol{\theta})\boldsymbol{\rho}\mathbf{U}(\boldsymbol{\theta})^\dagger\mathbf{M})$.

III. Training Landscape of Quantum Supervised Learning

The success of parameterized machine learning models depend heavily on their trainability. An principled way to characterize the trainability in the classical literature is to study the optimization landscape. For example, [18]–[22] show that many spurious local minima may exist in ClaNNs; [13], [23]–[26] shows that for certain designs of neural networks, the landscape can be benign.

On the quantum side, in [10], we conduct a quantitative investigation on the training landscape for QNNs for square loss function. Their result suggests that almost all under-parameterized QNNs can be hard to train, highlighting the necessity of adapting QNN designs to specific learning problem, or devising training algorithms beyond gradient-descent.

Training Landscape of QNNs. A common practice to perform supervised learning is through empirical risk minimization (ERM), i.e. finding a set of parameters that best aligns with the training sample with respect to a specific loss function l. Under the framework of ERM, training with the quantum dataset $\mathcal{S} = \{(\boldsymbol{\rho}_i, y_i)\}_{i=1}^m$ involves minimizing

$$L(\boldsymbol{\theta}; \mathcal{S}) := \frac{1}{m}\sum_{i=1}^m \Big(\operatorname{tr}(\mathbf{U}(\boldsymbol{\theta})\boldsymbol{\rho}_i\mathbf{U}^\dagger(\boldsymbol{\theta})\mathbf{M}) - y_i\Big)^2. \tag{3}$$

$\boldsymbol{\theta}$ is a local minimum if and only if there is an open set U containing $\boldsymbol{\theta}^*$ such that $L(\boldsymbol{\theta}^*; \mathcal{S}) \leq L(\boldsymbol{\theta}; \mathcal{S})$ for all $\boldsymbol{\theta} \in U$. A local minimum is global if the minimum value of $L(\cdot; \mathcal{S})$ is attained at θ^*.

Construction of Hard Datasets. The first result in [10] reveals that hard datasets can be constructed to introduce exponentially many spurious local minima in number of parameters for almost all QNNs with respect to the following measure:

$$\mathbf{U}(\boldsymbol{\theta}) = e^{-i\theta_p\mathbf{W}_p\mathbf{H}\mathbf{W}_p^\dagger}\cdots e^{-i\theta_1\mathbf{W}_1\mathbf{H}\mathbf{W}_1^\dagger} \tag{4}$$

where $\{\mathbf{W}_l\}_{l=1}^p$ are sampled independently with respect to the Haar measure on the d-dimensional unitary group $U(d)$, and $\mathbf{H}$ is a d-dimensional trace-0 Hermitian with eigenvalues ± 1. Up to a unitary transformation, the random model is equivalent to a circuit with p interleaving parameterized gate $\{e^{-i\theta_l\mathbf{H}}\}_{l=1}^p$ and unitary $\{\tilde{\mathbf{W}}_l\}_{l=1}^p$ randomly sampled with respect to the Haar measure; and any p-parameter QNN generated by two-level Hamiltonians can be expressed in Eqn. (4).

The parameterized operator $\exp(-i\theta_l\mathbf{W}_l\mathbf{H}_l\mathbf{W}_l^\dagger)$ contains terms proportional to $\cos\theta_l$ and $\sin\theta_l$. The measurement in the Heisenberg's picture $\mathbf{U}(\boldsymbol{\theta})^\dagger\mathbf{M}\mathbf{U}(\boldsymbol{\theta})$ can therefore be expanded in the Fourier basis as

$$\sum_{\boldsymbol{\xi}\in\{0,1,2\}^p} \Phi_{\boldsymbol{\xi}}(\mathbf{M}) \prod_{l:\xi_l=1} \cos 2\theta_l \prod_{l':\xi_{l'}=2} \sin 2\theta_{l'}. \tag{5}$$

The following theorem identifies a sufficient condition for the existence of hard datasets in terms of the Fourier components $\{\Phi_{\boldsymbol{\xi}}(\mathbf{M})\}$, which holds with high probability:

Theorem 3.1 (Construction: exponentially many local minima): Consider a random QNN sampled with respect to the measure defined in Eqn (4) with $p = O(\log(d))$. Then with probability $\geq 1 - O(d^{-1})$, (1) $\{\Phi_{\boldsymbol{\xi}}(\mathbf{M})\}_{\boldsymbol{\xi}\in\{0,1,2\}^p, \boldsymbol{\xi}\neq\mathbf{0}}$ forms a linearly independent set, and (2) a dataset $\mathcal{S}$ can be constructed to induce a loss function $L(\boldsymbol{\theta}; \mathcal{S})$ with 2^p local minima within each period, and $2^p - 1$ of these minima are spurious with positive suboptimality gap.

The construction relies on the classical idea of symmetry breaking, which involves constructing a training set $\mathcal{S}_0$ with $L(\boldsymbol{\theta}, \mathcal{S}_0)$ invariant under the translation $\theta_l \mapsto \theta_l + \frac{\pi}{2}$ and an asymmetric training set $\mathcal{S}_1$ that breaks the symmetry to create the suboptimal gaps. Concrete constructions of $\mathcal{S}_0$ and $\mathcal{S}_1$ are determined by linear systems, the solvability of which are guaranteed by the linear independence of $\{\Phi_{\boldsymbol{\xi}}(\mathbf{M})\}_{\boldsymbol{\xi}\in\{0,1,2\}^p, \boldsymbol{\xi}\neq\mathbf{0}}$.

The linear independence of the Fourier components follows from the positve definiteness of the Gram matrix $\left[G\right]_{\boldsymbol{\xi},\boldsymbol{\xi}'} := \operatorname{tr}(\Phi_{\boldsymbol{\xi}}(\mathbf{M})\Phi_{\boldsymbol{\xi}'}(\mathbf{M}))$. By estimating the moments of the matrix elements, the smallest eigenvalue of the Gram matrix G is shown to be positive with probability $\geq 1 - O(e^{-p})$ for dimension $d : \log(d) = \Theta(p)$, and hence the linear independence.

Upper Bound for Number of Local Minima. The second result in [10] states that the construction above is almost optimal in number of local minima:

Theorem 3.2 (Upper bound: the number of local minima): Consider non-degenerated QNNs composed of unitaries generated by two-level Hamiltonians $\{H_l\}_{l=1}^p$ with p parameters. For training set $\mathcal{S}$, within each period, the loss function $L(\boldsymbol{\theta};\mathcal{S})$ possesses at most $(4p)^p$ (strict) local minima.

This implies that the number of local minima in QNNs is upper-bounded, separating the landscape of QNNs from that of ClaNNs: The classical work of [18] shows that $\lfloor m/p \rfloor^p$ local minima can be constructed for ClaNNs with m training samples, which is unbounded as m grows.

The proof of Theorem 3.2 follows by bounding the number of local minima with the number of roots to a polynomial system: for arbitrary choice of two-level $\{\mathbf{H}_l\}_{l=1}^p$, observable $\mathbf{M}$ and training set $\mathcal{S}$, the support K of the Fourier spectrum of the loss function $L(\boldsymbol{\theta};\mathcal{S})$ is bounded in ℓ_1-norm: $\max_{\mathbf{k}\in K}\sum_{l=1}^p |k_l| \leq 2p$. Under the change of variable $c_l = \cos(\theta_l/T_l)$, $s_l = \sin(\theta_l/T_l)$, $\forall l \in [p]$, the number of local minima for the training loss function can be upper-bounded by the number of (isolated) solutions to a p-variable polynomial system with degree $2p$.

Empirical Evaluation. Constructions with 8 - 17 parameters are trained with gradient-based methods (Adam, RMSProp and L-BFGS), each with 5000 random initializations. It is observed in the numerical simulation, that for all three optimizers, the success rate for finding the global minima decay exponentially as the number of parameters increases. This indicates that the constructed spurious local minima indeed make it hard to optimize with gradient-based methods.

IV. Quantum Generative Adversarial Networks

The applicability of a model of machine learning depends on its capacity to accurately capture functions with required properties, as well as the robustness and scalability of training its parameters to attain those properties. An example of these issues can be seen in the application of quantum machine learning to generative learning. Generative learning is the task of learning a procedure to generate samples from some unknown distribution $\mathcal{D}$. Specifically, we learn a *generator* function G such that samples z from some latent distribution $\mathcal{Z}$ are mapped to samples $G(z)$ from the generated distribution. Generative Adversarial Networks (GANs) [27] have emerged as a powerful method of using ClaNNs to perform deep generative learning. In a GAN, an additional function known as a discriminator is introduced that attempts to distinguish the generated (*"fake"*) samples from (*true*) samples drawn from the real distribution. The training of such a system is modeled as an adversarial game; where the discriminator is trained to maximally differentiate real and fake samples, while the generator is simultaneously trained to minimize the difference. At equilibrium, the generated samples are distributed similarly to the target distribution.

The success of classical GANs raise the natural question of whether meaningful quantum counterparts exist. Quantum GANs could provide potential quantum speedups due to the fact that parameterized quantum circuits (in the generator and discriminator) cannot be efficiently simulated by classical models. There might therefore exist distributions that can be learned by an efficient quantum generator while an equally good classical generator may be infeasible to train or sample from. Since the seminal work of [8], there were several following proposals [28]–[34] of constructions of quantum GANs to generate quantum or classical data. Proof-of-principle practical demonstrations have also been achieved on a physical quantum computer [32], [34]. Many of these proposals for quantum GANs focused on using quantum generators to generate classical data. There are however many applications such as the investigation of quantum systems in condensed matter physics or quantum chemistry, which require the ability to generate quantum data, and it is often not natural to capture these learning tasks with the same loss functions as used in classical GANs. Furthermore, the training of classical GANs is notoriously delicate, and this phenomenon was strongly manifested in the quantum GAN proposals that did learn quantum data [28], [29]. Classically, the training behavior is often seen to depend on the metric used to capture the distance between real and fake distributions [35]. Widely used metrics such as the Kullback-Leibler (KL) divergence, Jensen-Shannon (JS) divergence, and total variational distance, can fail to be differentiable or even continuous in the parameters of the generator, when learning distributions with low dimensional support. A Quantum Wasserstein GAN (qWGAN) has been proposed by [11] to improve the robustness of learning quantum data, using a Wasserstein-type metric to capture the distance between distributions. The Wasserstein (or optimal transport) metric was proposed by the seminal work [35] to improve robustness for GAN training. This metric is appealing from the perspective of optimization due to its smoothness in terms of the training parameters of the generated distribution. A semi-metric for quantum states inspired by the Wasserstein distance that retains these smoothness properties is formulated in [11]. This is then used to construct a robustly trainable architecture for a GAN.

Quantum Wasserstein Semi-metric. The classical Wasserstein distance between distributions P, Q over $\mathcal{X}, \mathcal{Y}$ respectively can be defined for any cost function $c\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{R}$ as the minimum expected cost for any distribution over $\mathcal{X}\times\mathcal{Y}$ that has P, Q as its marginals. Classical distributions can be represented by diagonal density matrices, and classical cost functions by diagonal Hermitian matrices. The marginal of a distribution over $\mathcal{X}\times\mathcal{Y}$ on $\mathcal{X}$ or $\mathcal{Y}$ is evaluated by taking the partial trace over $\mathcal{Y}$ or $\mathcal{X}$ respectively. In density matrix form, the distance is defined as

$$\mathcal{W}(P,Q) = \min_{\pi} \operatorname{tr}(\pi C) \tag{6}$$

$$\text{s.t. } \pi \in \mathcal{D}(\mathcal{X}\times\mathcal{Y}) \tag{7}$$

$$\operatorname{tr}_{\mathcal{Y}} \pi = \operatorname{diag}(P), \operatorname{tr}_{\mathcal{X}} \pi = \operatorname{diag}(Q), \tag{8}$$

$$C = \operatorname{diag}(c(x,y)) \tag{9}$$

Quantum distributions (or quantum states) are already density matrices so the diagonality restrictions on the marginals as well as C can be removed. With the additional restriction that identical states are at distance 0 from each other, it can be shown that in the quantum case C must be fixed to $C_0 = (I + \mathrm{SWAP}_{\mathcal{X}\times\mathcal{Y}})/2$ where SWAP is defined to be the unique unitary operation such that $\mathrm{SWAP}(\mathbf{x} \otimes \mathbf{y}) = (\mathbf{y} \otimes \mathbf{x})$ for all $\mathbf{x} \in \mathcal{C}^{\mathcal{X}}, \mathbf{y} \in \mathcal{C}^{\mathcal{Y}}$. The quantum Wasserstein semi-metric is then defined as

$$\min_{\pi} \mathrm{tr}(\pi C_0) = \frac{1}{2}\mathrm{tr}(\pi(I + \mathrm{SWAP})) \tag{10}$$

$$\text{s.t. } \pi \in \mathcal{D}(\mathcal{X}\times\mathcal{Y}), \mathrm{tr}_{\mathcal{Y}}\,\pi = P, \mathrm{tr}_{\mathcal{X}}\,\pi = Q \tag{11}$$

Regularization and Construction of GAN. The optimization problem (10) is difficult to solve when the underlying distributions are parameterized due to the presence of hard semidefinite constraints (which can become non-convex in the parameters). Classically [36] strongly convex regularizers can be added to the objective of a primal optimization problem to remove hard constraints in the dual. Using the quantum relative entropy as a regularizer for the quantum-Wasserstein semimetric results in the optimization problem

$$\min_{\pi} \mathrm{Tr}(\pi C_0) + \lambda \mathrm{Tr}(\pi\log(\pi) - \pi\log(P\otimes Q)) \tag{12}$$

$$\text{s.t. } \pi \in \mathcal{D}(\mathcal{X}\times\mathcal{Y}), \mathrm{tr}_{\mathcal{Y}}\,\pi = P, \mathrm{tr}_{\mathcal{X}}\,\pi = Q \tag{13}$$

By SDP-duality and employing the Golden-Thompson trace inequality the above optimization problem is shown to be upper bounded by

$$\mathcal{QW}(P,Q) = \max_{\phi,\psi} \mathcal{E}_Q[\psi] - \mathcal{E}_P[\phi] - \mathcal{E}_{P\otimes Q}[\xi_R] \tag{14}$$

$$\text{s.t.} \quad \phi \in \mathcal{H}(\mathcal{X}), \psi \in \mathcal{H}(\mathcal{Y}) \tag{15}$$

where ξ_R is a regularizing Hermitian:

$$\xi_R = \frac{\lambda}{e}\exp\left(\frac{-C - \phi\otimes I_{\mathcal{Y}} + I_{\mathcal{X}}\otimes\psi}{\lambda}\right) \tag{16}$$

A quantum Wasserstein GAN learning state Q formulates the generated state as $G(\boldsymbol{\theta_1})\boldsymbol{\rho}_0 G(\boldsymbol{\theta_1})^\dagger$ where $\boldsymbol{\rho}_0$ is some fixed starting state, and G is a parameterized This results in the adversarial game

$$\min_{\theta_1} \max_{\phi\in\mathcal{H}(\mathcal{X}),\psi\in\mathcal{H}(\mathcal{Y})} \mathcal{E}_Q[\psi] - \mathcal{E}_{G(\boldsymbol{\theta_1})\boldsymbol{\rho}_0 G(\boldsymbol{\theta_1})^\dagger}[\phi] - \mathcal{E}_{G(\boldsymbol{\theta_1})\boldsymbol{\rho}_0 G(\boldsymbol{\theta_1})^\dagger\otimes Q}[\xi_R] \tag{17}$$

Hermitian matrices can be parameterized using parameterized quantum circuits by defining $\phi = U(\boldsymbol{\theta}_2)H_x U(\boldsymbol{\theta}_2)^\dagger, \psi = V(\boldsymbol{\theta}_3)H_y V(\boldsymbol{\theta}_3)^\dagger$, where H_x, H_y are diagonal measurements that can be easily performed. This yields an adversarial unconstrained optimization problem in $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2.\boldsymbol{\theta}_3$. Due to the formulation of the quantum Wasserstein semi-metric the objective can be shown to be continuous and differentiable in the real parameters [11].

Empirical Evaluation. The qWGAN has been used to learn pure quantum states with 1,2,4 and 8 qubits, as well as mixed states with 1,2 and 3 qubits. Convergence to 99% fidelity is observed for pure states in ~ 150 iterations for 1 and 2 qubit states, ~ 400 iterations for 4 qubit states, and ~ 800 iterations for 8 qubit states. For mixed states, we observe convergence in ~ 200 iterations for $1, 2$ qubit states, and ~ 600 iterations for 3 qubit states. In comparison, previous approaches for learning quantum data either only demonstrate learning 1 qubit pure and mixed states [28] (converging after ~ 150 iterations) or show that gradient based methods no longer make progress beyond 6 qubit pure states [29]. In contrast the qWGAN can be trained to learn 8 qubit states using simple gradient descent.

Circuit Compression. A quantum generative model can be used to compress a large circuit $U \colon \mathcal{C}^{\mathcal{X}} \times \mathcal{C}^{\mathcal{Y}}$ by training a smaller variational ansatz G so that $(I\otimes G)\boldsymbol{\Phi}\boldsymbol{\Phi}^\dagger(I\otimes G)^\dagger$ approximates $(I\otimes U)\boldsymbol{\Phi}\boldsymbol{\Phi}^\dagger(I\otimes U)^\dagger$ where $\boldsymbol{\Phi} = \sum_{\mathbf{x}\in\mathcal{X}} \mathbf{x}\otimes\mathbf{x}$. The qWGAN has been used to compress a circuit simulating the 1d 3-qubit Heisenberg model: the original circuit that requires ~ 11900 gates via the best known theoretical bounds is fitted to 99% fidelity using an ansatz with 52 gates, indicating that for a large majority of inputs the smaller circuit effectively reproduces the larger one.

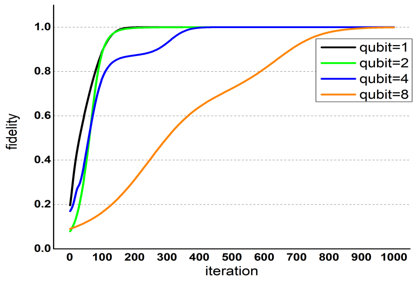

Fig. 2. Learning pure state with qWGAN

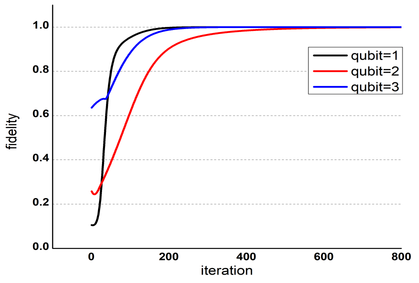

Fig. 3. Learning mixed states with qWGAN

V. Differentiable Quantum Programming Languages

The predicted future applications of variational quantum systems and QNNs has highlighted the importance of a scalable and convenient system for implementing them programatically. The typical practice for ClaNNs is to encode their evaluation in computation graphs so that the gradients of the system can be efficiently computed. Gradients for possibly complicated systems can be time consuming to program from scratch (even when efficient schemes are known) and their implementations introduce an additional source of bugs. Popular auto-differentiation packages such as Autograd and JAX [37] automate the process of generating procedures to compute the gradients given the source code of a parameterized function.

However, computing these gradients of loss functions from variational quantum circuits has a similar complexity of simulating quantum circuits, which is infeasible for classical means. Thus, the ability of evaluating "quantum" gradients efficiently by quantum means is critical for the scalability of quantum variational methods.

Fortunately, there has been progress towards similar procedures for quantum circuits. Mitarai et. al [14] and Schuld et. al [38] introduced and refined a *parameter shift rule* that modifies a parameterized circuit to obtain circuits computing the partial derivatives of the loss function. Libraries for quantum programming such as Pennylane [39] and TensorFlow Quantum [40] use these methods to autodifferentiate computation graphs consisting of quantum nodes representing individual parameterized quantum circuits.

The above approach, however, does not handle general quantum programs that may have conditional statements or loops conditioned on registers representing quantum states. Such programs are interesting for two reasons: (1) Dynamic control flow has proved advantageous [41], [42] in several classical learning systems, and (2) loops and conditional statements can allow more complex programs to be implemented using a smaller number of quantum registers which has particular advantages for near-term noisy hardware implementations. Hardware providers have begun to natively provide the option for mid-circuit measurement and reuse of quantum registers [43], making it likely that such programs can be natively compiled without inflating the required resources.

A general auto-differentiation scheme for quantum programs, called differentiable quantum programming languages, has been formulated by Zhu et. al [12]. In particular, Zhu et. al provide a rigorous formalization of the auto-differentiation technique applied to quantum circuits, which includes a formal formulation of quantum programs, their semantics, and the meaning of differentiation of them. They also design the code-transformation rules for auto differentiation of general quantum programs and prove their correctness.

An implementation of these rules is publicly available[1] and has been used to perform the following case study which demonstrates the effectiveness of using quantum control in a resource constrained variational system.

The case study compares two 4-qubit QNNs to solve a simple classification problem over 4-bit inputs $z = z_1 z_2 z_3 z_4 \in \{0,1\}^4$ with true label given by $f(z) = \neg(z_1 \oplus z_4)$. Two 4-qubit QNNss P_1 (**no control**) and P_2 (**with control**) are constructed that each consists of a single-qubit Pauli rotation gate on each qubit. For parameters $\Gamma = \{\gamma_1, \ldots, \gamma_{12}\}$ define the program

$$\begin{aligned} Q(\Gamma) \equiv \quad & R_{\mathbf{X}}(\gamma_1)[q_1]; R_{\mathbf{X}}(\gamma_2)[q_2]; R_{\mathbf{X}}(\gamma_3)[q_3]; R_{\mathbf{X}}(\gamma_4)[q_4]; \\ & R_{\mathbf{Y}}(\gamma_5)[q_1]; R_{\mathbf{Y}}(\gamma_6)[q_2]; R_{\mathbf{Y}}(\gamma_7)[q_3]; R_{\mathbf{Y}}(\gamma_8)[q_4]; \\ & R_{\mathbf{Z}}(\gamma_9)[q_1]; R_{\mathbf{Z}}(\gamma_{10})[q_2]; R_{\mathbf{Z}}(\gamma_{11})[q_3]; R_{\mathbf{Z}}(\gamma_{12})[q_4], \end{aligned}$$

[1]https://github.com/LibertasSpZ/adcompile.

where q_1, q_2, q_3, q_4 refer to 4 single qubit registers and $R_\sigma(\theta) = \exp(i\sigma\theta/2)$ whenever σ is a Pauli matrix. Given parameters $\Theta = \{\theta_1, \ldots, \theta_{12}\}, \Phi = \{\phi_1, \ldots, \phi_{12}\}$, define

$$P_1(\Theta, \Phi) \equiv Q(\Theta); Q(\Phi). \tag{18}$$

Similarly, for parameters $\Theta = \{\theta_1, \ldots, \theta_{12}\}, \Phi = \{\phi_1, \ldots, \phi_{12}\}$, $\Psi = \{\psi_1, \ldots, \psi_{12}\}$, define

$$\begin{aligned} P_2(\Theta, \Phi, \Psi) \equiv Q(\Theta); \textbf{case } M[q_1] = 0 \rightarrow \quad & Q(\Phi) \\ 1 \rightarrow \quad & Q(\Psi). \end{aligned} \tag{19}$$

Note that P_1 and P_2 require the same number of qubits and quantum gates for each single run. To use a program P_i for training or classification, q_1, q_2, q_3, q_4 are initialized to the classical feature vector $z = z_1, z_2, z_3, z_4$ and then execute the program. The predicted label is given by measuring the fourth qubit q_4 in the 0/1 basis. Learning is performed by minimizing a loss function given by the squared loss function over all inputs, treating the expectation value of the output as the output of the circuit. P_1 is trained using Pennylane and P_2 is trained using gradient descent where the gradient is computed according to the transformation rules [12]. It is observed that after 1000 epochs with manually optimized hyperparameters, the loss for P_1 (**no control**) attains a minimum of 0.6931 in less than 100 epochs and subsequently plateaus. The loss for P_2 (**with control**) continues to decrease and attains a minimum of 0.0936.

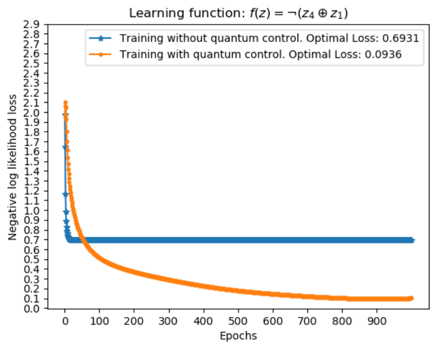

Fig. 4. Training P_1 and P_2 to classify inputs according to the labelling function $f(z) = \neg(z_1 \oplus z_4)$.

Acknowledgment

We would like to thank the invitation from the organizers of the ICCAD special session in quantum machine learning. We also want to acknowledge our co-authors (Yiming Huang, Tongyang Li, Soheil Feizi, Shaopeng Zhu, Shih-Han Hung) for their insights and contributions in highlighted research results.

References

[1] F. Arute, K. Arya, R. Babbush, *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[2] H.-S. Zhong, H. Wang, Y.-H. Deng, *et al.*, "Quantum computational advantage using photons," *Science*, vol. 370, no. 6523, pp. 1460–1463, 2020, ISSN: 0036-8075. DOI: 10.1126/science.abe8770. eprint: https://science.sciencemag.org/content/370/6523/1460.full.pdf. [Online]. Available: https://science.sciencemag.org/content/370/6523/1460.

[3] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, no. 79, 2018.

[4] A. Peruzzo, J. McClean, P. Shadbolt, *et al.*, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, no. 1, p. 4213, 2014.

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

[6] E. Farhi, H. Neven, *et al.*, "Classification with quantum neural networks on near term processors," *Quantum Review Letters*, vol. 1, no. 2 (2020), pp. 10–37 686, 2020.

[7] M. Benedetti, E. Lloyd, S. Sack, *et al.*, "Parameterized quantum circuits as machine learning models," *Quantum Science and Technology*, vol. 4, no. 4, p. 043 001, 2019.

[8] S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," *Physical Review Letters*, vol. 121, p. 040 502, 4 2018. eprint: arXiv:1804.09139.

[9] J. R. McClean, S. Boixo, V. N. Smelyanskiy, *et al.*, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, no. 1, p. 4812, 2018.

[10] X. You and X. Wu, "Exponentially many local minima in quantum neural networks," in *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[11] S. Chakrabarti, H. Yiming, T. Li, *et al.*, "Quantum wasserstein generative adversarial networks," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, *et al.*, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/f35fd567065af297ae65b621e0a21ae9-Paper.pdf.

[12] S. Zhu, S.-H. Hung, S. Chakrabarti, *et al.*, "On the principles of differentiable quantum programming languages," in *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI 2020, London, UK: Association for Computing Machinery, 2020, pp. 272–285, ISBN: 9781450376136. DOI: 10.1145/3385412.3386011. [Online]. Available: https://doi.org/10.1145/3385412.3386011.

[13] K. Kawaguchi, "Deep learning without poor local minima," in *Advances in neural information processing systems*, 2016, pp. 586–594.

[14] K. Mitarai, M. Negoro, M. Kitagawa, *et al.*, "Quantum circuit learning," *Physical Review A*, vol. 98, no. 3, p. 032 309, 2018.

[15] S. Lloyd, M. Schuld, A. Ijaz, *et al.*, "Quantum embeddings for machine learning," *arXiv preprint arXiv:2001.03622*, 2020.

[16] J. Li, X. Yang, X. Peng, *et al.*, "Hybrid quantum-classical approach to quantum optimal control," *Physical review letters*, vol. 118, no. 15, p. 150 503, 2017.

[17] M. Ostaszewski, E. Grant, and M. Benedetti, "Quantum circuit structure learning," *arXiv preprint arXiv:1905.09692*, 2019.

[18] P. Auer, M. Herbster, and M. K. Warmuth, "Exponentially many local minima for single neurons," in *Advances in neural information processing systems*, 1996, pp. 316–322.

[19] I. Safran and O. Shamir, "Spurious local minima are common in two-layer relu neural networks," in *International Conference on Machine Learning*, PMLR, 2018, pp. 4433–4441.

[20] C. Yun, S. Sra, and A. Jadbabaie, "Small nonlinearities in activation functions create bad local minima in neural networks," *arXiv preprint arXiv:1802.03487*, 2018.

[21] T. Ding, D. Li, and R. Sun, "Sub-optimal local minima exist for almost all over-parameterized neural networks," *arXiv preprint arXiv:1911.01413*, 2019.

[22] L. Venturi, A. S. Bandeira, and J. Bruna, "Spurious valleys in two-layer neural network optimization landscapes," *arXiv preprint arXiv:1802.06384*, 2018.

[23] S. Du and J. Lee, "On the power of over-parametrization in neural networks with quadratic activation," in *International Conference on Machine Learning*, PMLR, 2018, pp. 1329–1338.

[24] D. Soudry and Y. Carmon, "No bad local minima: Data independent training error guarantees for multilayer neural networks," *arXiv preprint arXiv:1605.08361*, 2016.

[25] Q. Nguyen and M. Hein, "The loss surface of deep and wide neural networks," in *International conference on machine learning*, PMLR, 2017, pp. 2603–2612.

[26] D. Li, T. Ding, and R. Sun, "Over-parameterized deep neural networks have no strict local minima for any continuous activations," *arXiv preprint arXiv:1812.11039*, 2018.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680. eprint: arXiv:1406.2661.

[28] P.-L. Dallaire-Demers and N. Killoran, "Quantum generative adversarial networks," *Physical Review A*, vol. 98, p. 012 324, 1 Jul. 2018. eprint: arXiv:1804.08641.

[29] M. Benedetti, E. Grant, L. Wossnig, *et al.*, "Adversarial quantum circuit learning for pure state approximation," *New Journal of Physics*, vol. 21, no. 4, p. 043 023, 2019. eprint: arXiv:1806.00463.

[30] J. Zeng, Y. Wu, J.-G. Liu, *et al.*, "Learning and inference on generative adversarial quantum circuits," 2018.

[31] H. Situ, Z. He, L. Li, *et al.*, "Quantum generative adversarial network for generating discrete data," 2018.

[32] L. Hu, S.-H. Wu, W. Cai, *et al.*, "Quantum generative adversarial learning in a superconducting quantum circuit," *Science advances*, vol. 5, no. 1, eaav2761, 2019. eprint: arXiv:1808.02893.

[33] J. Romero and A. Aspuru-Guzik, "Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions," 2019.

[34] C. Zoufal, A. Lucchi, and S. Woerner, "Quantum generative adversarial networks for learning and loading random distributions," 2019.

[35] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223. eprint: arXiv:1701.07875.

[36] M. Sanjabi, J. Ba, M. Razaviyayn, *et al.*, "On the convergence and robustness of training GANs with regularized optimal transport," in *Advances in Neural Information Processing Systems 31*, Curran Associates, Inc., 2018, pp. 7091–7101. eprint: arXiv:1802.08249.

[37] J. Bradbury, R. Frostig, P. Hawkins, *et al.*, *JAX: Composable transformations of Python+NumPy programs*, version 0.2.5, 2018. [Online]. Available: http://github.com/google/jax.

[38] M. Schuld, V. Bergholm, C. Gogolin, *et al.*, "Evaluating analytic gradients on quantum hardware," *Physical Review A*, vol. 99, no. 3, p. 032 331, 2019. eprint: arXiv:1811.11184.

[39] V. Bergholm, J. Izaac, M. Schuld, *et al.*, "PennyLane: Automatic differentiation of hybrid quantum-classical computations," 2018.

[40] M. Broughton, G. Verdon, T. McCourt, *et al.*, *Tensorflow quantum: A software framework for quantum machine learning*, 2021. arXiv: 2003.02989 [quant-ph].

[41] A. Graves, G. Wayne, M. Reynolds, *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, vol. 538, p. 471, Oct. 2016.

[42] E. Grefenstette, K. M. Hermann, M. Suleyman, *et al.*, "Learning to transduce with unbounded memory," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15, Montreal, Canada: MIT Press, 2015, pp. 1828–1836. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969442.2969444.

[43] J. M. Pino, J. M. Dreiling, C. Figgatt, *et al.*, "Demonstration of the trapped-ion quantum ccd computer architecture," *Nature*, vol. 592, no. 7853, pp. 209–213, Apr. 2021, ISSN: 1476-4687. DOI: 10.1038/s41586-021-03318-4. [Online]. Available: http://dx.doi.org/10.1038/s41586-021-03318-4.