Variational Quantum Classifiers Through the Lens of the Hessian

Pinaki Sen[1] and Amandeep Singh Bhatia[2*]

[1]*National Institute of Technology, Agartala, Tripura, India*
[2]*Chitkara University Institute of Engineering & Technology, Chitkara University, Punjab, India*
E-mail: [1]pinakisen.eenita@gmail.com, [2]amandeepbhatia.singh@gmail.com

In quantum computing, the variational quantum algorithms (VQAs) are well suited for finding optimal combinations of things in specific applications ranging from chemistry all the way to finance. The training of VQAs with gradient descent optimization algorithm has shown a good convergence. At an early stage, the simulation of variational quantum circuits on noisy intermediate-scale quantum (NISQ) devices suffers from noisy outputs. Just like classical deep learning, it also suffers from vanishing gradient problems. It is a realistic goal to study the topology of loss landscape, to visualize the curvature information and trainability of these circuits in the existence of vanishing gradients. In this paper, we calculated the Hessian and visualized the loss landscape of variational quantum classifiers at different points in parameter space. The curvature information of variational quantum classifiers (VQC) is interpreted and the loss function's convergence is shown. It helps us better understand the behavior of variational quantum circuits to tackle optimization problems efficiently. We investigated the variational quantum classifiers via Hessian on quantum computers, started with a simple 4-bit parity problem to gain insight into the practical behavior of Hessian, then thoroughly analyzed the behavior of Hessian's eigenvalues on training the variational quantum classifier for the Diabetes dataset. Finally, we show that how the adaptive Hessian learning rate can influence the convergence while training the variational circuits.

I.   INTRODUCTION & MOTIVATION

In recent years, the enhancement of machine learning algorithms by noisy intermediate-scale quantum (NISQ) technology and mainly the variational quantum circuits have garnered significant attention among academic and research communities [1]. The researchers have applied the variational quantum algorithms in various applications in the NISQ era, mainly the ones related to quantum artificial intelligence. The variational quantum algorithms (VQAs) have shown great learning capability to counterbalance the errors in the device framework. They are considered to be the greatest hope for the journey toward quantum advantage. The first variational quantum eigensolver (VQE) was proposed as a state ansatz to determine the ground state energy of physical systems [2]. Since the first VQE introduced, several VQE variants have been proposed with the plethora of alterations for computation of excited states such as orthogonality constrained VQE [3], subspace approach VQE [4, 5], adiabatically assisted VQE [6] and multistate contracted VQE [7]. These variational quantum circuits are constructed with an ansatz whose parameters are trained with gradient-based methods to minimize the cost. For combinatorial optimization tasks, the quantum alternating operator ansatz (QAOA) was proposed originally to attain approximate solutions [8]. Later on, it was used as a standalone ansatz. These architectures of variational quantum circuits have shown to be computationally universal [9].

Recently, the use of VQAs for solving a bunch of real-world problems has got an incredible response and is widely applied in quantum machine learning applications. VQAs consist of a small number of qubits and quantum circuits, which make them resistant to noise. The VQAs are effective for a classification task in machine learning; the objective is to train a classifier and predict the label of each input accurately [10, 45]. Suppose, a training data is given $\{x_i, y_i\}$, where x_i's and y_i's are inputs and labels respectively. The variational quantum circuit is used as a black box to predict the right label (y_i) for each input after embedding the classical data into the quantum states. VQAs are the quantum variants of neural networks, the most commonly used and highly successful machine learning model. Till now, several architectures for quantum neural networks (QNNs) have been proposed and applied in different areas [11–14]. Recently, Cong et al. [15] introduced the quantum convolutional neural networks (QCNNs) and used them to discriminate quantum states of distinct topological phases. Bhatia et al. [46] performed the simulation of several entangled states on a quantum computer. Romero et al. [16] proposed a quantum variational autoencoder for compressing the quantum data efficiently. Pepper et al. [17] demonstrated the experimental realization of a proposed quantum autoencoder, which will likely be an essential primordial in quantum machine learning. In recent years, generative adversarial networks (GANs) have been an exciting topic of research in classical machine learning. Romero and Aspuru-Guzik [18] introduced the quantum variant of GAN for learning continuous distributions and to speedup
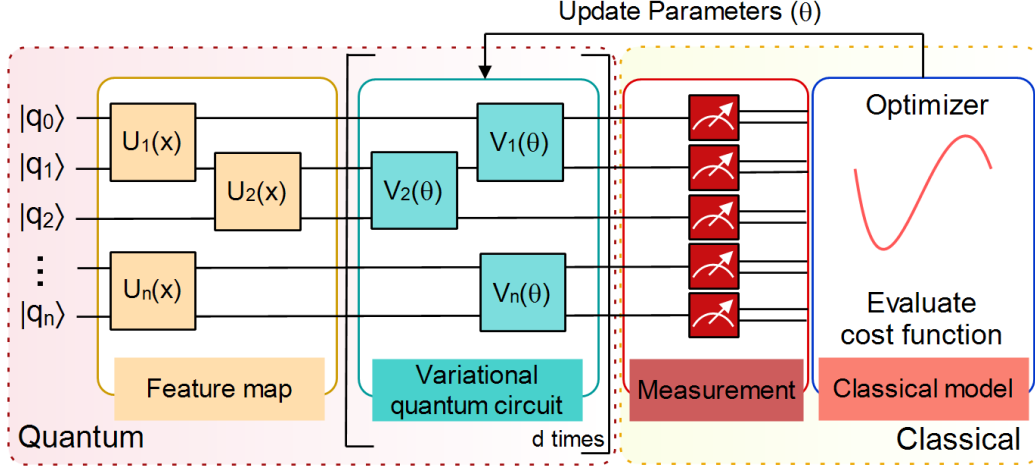
classical GAN using quantum systems.

FIG. 1: **Schematic representation of a variational quantum circuit (VQC).** VQC is created by combining the feature map, variational circuit and measurement component. It is followed by classical optimization model. It includes the iterative implementation of quantum and classical components of the circuit.

The major downside of variational quantum circuits is the occurrence of a barren plateau problem, which vanishes the gradients of cost function exponentially with the increase in the number of qubits [19, 20]. It can be represented by flat plateaus of the loss function [21]. Firstly, in 2018, McClean et al. [22] studied the barren plateau phenomenon numerically for a class of random quantum circuits. It has shown that variance of the gradients of modest size quantum circuits vanishes exponentially. Cerezo et al. [23] described a barren plateau phenomenon as a point in the parameter space and investigated it for shallow quantum neural networks (QNNs). It has proved that global cost gradients vanish exponentially at all depths, whereas local costs exhibit non-vanishing gradients at shorter depth quantum circuits. Grant et al. [24] investigated the problem of a barren plateau of parametrized quantum circuits in the energy landscape. The approach is based on the initialization of parameters to avoid the barren plateaus. Initially, some parameter values are selected randomly, and the remaining values are chosen so that it becomes a chain of shallow circuits. The depth of circuits is used to determine the first updated parameter not to be lodged in a barren plateaus problem during training. Recently, it has investigated that the barren plateau is missing in QNNs and QCNNs with tree tensor network (TTN) architecture [25, 26].

In classical machine learning, the loss landscapes of neural networks and their characterization via the lens of Hessian have been well investigated. Several works have explored the flatness of local minima using the eigenvalues of Hessian [26–30]. During training, the generalization ability of the neural networks depends on batch size. It has explored that mini-batch size tends to favor flat minima basins of the loss function with several eigenvalues of the Hessian $\lambda_j = 0$ and hardly any $\lambda_j < 0$ [31]. Perez-Salinas et al. attained fast convergence for some quantum classification problems using Hessian based optimization method called batched optimization. Rebentrost et al. [32] proposed the quantum variants of two popular iterative optimization algorithms, Newton's and gradient descent. At time t, numerous copies of quantum state $|\psi\rangle_t$ are used to generate other numerous copies of quantum state $|\psi\rangle_{t'}$ at $t' = t+1$ by using the Hessian of an objective function and in consideration of a gradient vector. Recently, Huembeli and Dauphin [33] calculated the Hessian of a loss function and characterized the loss landscape of variational quantum circuits. It showed that the Hessian helps escape the flat surface of loss function for certain data-driven variational quantum circuits.

The VQCs have been extensively employed in a wide array of new applications. Fig 1 shows the schematic representation of a variational quantum circuit. It involves evaluating a cost function or its gradient on a quantum system [34, 35]. A classical optimization loop trains the parameters (θ) of a variational quantum circuit $V(\theta)$ to reduce the cost. It is well-known that the feature map encodings, variational parameterized circuit architecture designs produce loss functions that train easier, and well-selected training parameters "*optimizer*" that generalize well. Nevertheless, the effects of parameters on the enitre loss landscape are not well studied. It has not received the attention it deserves. The occurrence of barren plateaus issue could abolish the quantum advantage with a parametrized quantum circuit [36]. Hence, it is a natural goal to study the loss landscape of variational quantum classifiers with the eigenvalues of Hessian to recognize when the quantum speedup is achievable. The visualization of the loss landscape of data-driven variational quantum algorithms remains a vital and highly active area of research. In this paper, we visualized the loss landscape of variational quantum classifiers through the lens of Hessian of the loss function. The eigenvectors and eigenvalues of Hessian present a clear interpretation of the loss landscape of a VQC. We started with a 4-bit parity

problem to provide perception about the behavior of Hessian and then studied the VQC trained on diabetes classical data acting as a classifier.

The organization of the rest of this paper is as follows: Sect. 2 is devoted to preliminaries and Hessian's computation of variational quantum classifier. In Sect. 3, we computed the Hessian on a quantum simulator and visualized the curvature information of a parity function. In Sect. 3 (B), we characterized the loss landscape (i.e., curvature information) of data-driven variational quantum classifiers via Hessian of the loss function, and the experimental results are plotted for the diabetes dataset. In Sect. 4, we show how the adaptive Hessian learning rate can help to overshoot the cost that helps it avoid getting stuck in local minima during training of variational circuits. Finally, Sect. 5 is the conclusion.

II. PRELIMINARIES

In this Section, some basic concepts of loss function visualization with the Hessian matrix are given. Here, we give the background required to understand our results. Consider a real-valued function $f(\theta) = \theta_1, \theta_2, \ldots, \theta_n$. The Hessian matrix $(H^2 f(\theta))$ of $f(\theta)$ is represented as the square matrix of the second derivatives of a real-valued function of n variables, which help us to characterize the loss landscape (i.e. curvature information) [37, 38]. The gradient $(\bigtriangledown f(\theta))$ gives the partial derivatives of the function. The Hessian operator $H^2 f(\theta)$ gives the partial derivatives of the gradient [39].

$$\bigtriangledown f(\theta) = \begin{bmatrix} \dfrac{\partial f}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial f}{\partial \theta_n} \end{bmatrix}, H^2 f(\theta) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial \theta_1^2} & \cdots & \dfrac{\partial^2 f}{\partial \theta_1 \theta_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial \theta_n \theta_1} & \cdots & \dfrac{\partial^2 f}{\partial \theta_n^2} \end{bmatrix} \tag{1}$$

If the second derivatives are continuous, then the Hessian is symmetric. The Hessian matrix consists of information about the geometric information of the function. Its eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_n)$'s are real and used to get this curvature information. So the only thing to examine is whether the eigenvalues (λ_i)'s are negative or positive. Suppose, x_i is an eigenvector associated with (λ_i), then it is represented as i^{th} eigenpair (λ_i, x_i) of H. Using the Hessian matrix, we can determine whether the certain point θ on a surface is locally positive or negative. The eigenvalues of the Hessian matrix give the directions of the derivatives. Suppose the Hessian evaluated at a given point θ consists of all positive eigenvalues $(\lambda_i > 0)$ (i.e., positive-definite matrix). In that case, it shows a local positive curvature, and θ is a local minimum of f. Similarly, if all the eigenvalues are negative, it shows a locally negative curvature, and θ is a local maximum of f. The zero eigenvalues indicate the zero curvature of the function or flat directions. If the eigenvalues are mixed (some positive, some negative), then the surface has a saddle point θ of f. Thus, the Hessian can be used to determine the convexity and concavity of a function of one or two variables [40].

A. Hessian Computation of VQC

In classical machine learning, the neural networks are trained over a dataset consisting of feature vectors $\{x_i\}$ and labels $\{y_i\}$ by reducing the cost function as

$$C(\theta) = \frac{1}{n} \sum_{j=1}^{n} c(l(\theta, x_j), y_j), \tag{2}$$

where the $l(\theta, x_j)$ is the prediction parameterized by weights (θ) of the neural network, $c(.)$ denotes the loss function that measures how well the neural network predicts the label by calculating its difference with the neural network prediction, and n denotes the size of data samples. The loss functions exist in a high-dimensional space due to the presence of several parameters in neural networks. Although, its visualization is not possible in higher-dimensional space. Analogously, the loss landscape of VQC has not extensively examined as compared to classical neural networks. In a quantum layer, the classical data $(\overrightarrow{x})$ is encoded into the quantum state $|f(\theta, \overrightarrow{x})\rangle$ using a feature map consisting of quantum gates with parameters. In classical layer, the parameterized function $f(\theta, \overrightarrow{x})$ is evaluated depends upon the learning parameters (θ) in variational quantum circuit on performing measurement. The tuning of parameters (θ) is executed by minimizing a loss function on a classical computer.

The gradient of a quantum circuit can be evaluated by estimating the expectation value of an observable with reference to θ. It consists a series of unitary transformations. The gradient of an expectation value of an observable

$\langle f(\theta, \overrightarrow{x}) \rangle = \langle \overrightarrow{x} | V^+(\theta) f V(\theta) | \overrightarrow{x} \rangle$ can be given as

$$\frac{\partial \langle f(\theta, \overrightarrow{x}) \rangle}{\partial \theta_j} = \frac{1}{2} \Big[\Big\langle f\Big(\Big(\theta_{\overline{j}}, \theta_j + \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle - \Big\langle f\Big(\Big(\theta_{\overline{j}}, \theta_j - \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle \Big] \tag{3}$$

where $\theta_{\overline{j}}$ denotes the all parameters except θ_j, V is a product of unitary matrices, θ_i is an angle which parametrized the $V(\theta) = e^{-j\theta_j P_j/2}$, where P_j is a Hermitian operator with eigenvalues ± 1. Let us now define the Hessian matrix elements. The Hessian (H) of a quantum circuit can be computed by performing the parameter shift rule two times [33, 41–43].

$$H_{ij} = \frac{\partial^2 \langle f(\theta, \overrightarrow{x}) \rangle}{\partial \theta_i \partial \theta_j} = \partial_i \partial_j \langle f(\theta, \overrightarrow{x}) \rangle \tag{4}$$

$$\begin{aligned} H_{ij} = \frac{1}{4} \Big[& \Big\langle f\Big(\Big(\theta_{\overline{j},i}, \theta_j + \frac{\pi}{2}, \theta_i + \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle \\ & - \Big\langle f\Big(\Big(\theta_{\overline{j},i}, \theta_j + \frac{\pi}{2}, \theta_i - \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle \\ & - \Big\langle f\Big(\Big(\theta_{\overline{j},i}, \theta_j - \frac{\pi}{2}, \theta_i + \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle \\ & + \Big\langle f\Big(\Big(\theta_{\overline{j},i}, \theta_j - \frac{\pi}{2}, \theta_i - \frac{\pi}{2} \Big), \overrightarrow{x} \Big) \Big\rangle \Big] \end{aligned} \tag{5}$$

Thus, the loss function $c(f(\theta, \overrightarrow{x}))$ curvature can be studied via second-order derivative of the loss. The chain rule is performed twice to obtain the Hessian matrix elements of c.

$$\partial \theta_i \partial \theta_j c(f(\theta, \overrightarrow{x}), y) = \partial \theta_i \partial \theta_j f(\theta, \overrightarrow{x}) c'(f(\theta, \overrightarrow{x})) + \partial \theta_j \\ f(\theta, \overrightarrow{x}) \partial \theta_i f(\theta, \overrightarrow{x}) c''(f(\theta, \overrightarrow{x})) \tag{6}$$

where c' and c'' denote the first order and second order derivatives of the loss function, respectively with respect to the parameter (θ). In this paper, we considered the cost (or loss) function $c(f(\theta, \overrightarrow{x}), y) = (y - f(\theta, \overrightarrow{x}))^2$, which will be minimized. On calculating the derivatives, the Eq. (6) becomes:

$$\partial \theta_i \partial \theta_j c(f(\theta, \overrightarrow{x}), y) = \partial \theta_i \partial \theta_j f(\theta, \overrightarrow{x})(-2(y - f(\theta, \overrightarrow{x}))) \\ + \partial \theta_j f(\theta, \overrightarrow{x}) \partial \theta_j f(\theta, \overrightarrow{x}).2 \tag{7}$$

If the cost displays barren plateau, then its variance is vanishing exponentially as $Var[\partial_i f(\theta, \overrightarrow{x})] \leq F(n)$, where $F(n) \in \mathcal{O}(1/a^n)$, for $a > 1$ [22, 43, 44]. Then, the Hessian matrix elements are vanishing exponentially.

III. EXPERIMENT SETTINGS

We have used the PyTorch library [47], and pennylane package [48] for developing and training the variational quantum classifiers. The code is written in pennylane with a large number of parameters to expedite the experiments. The implementation is performed using PyTorch to accelerate the simulation using algebraic manipulation.

A. Warm-up Example

In this section, we characterized the loss landscape curvature of the four-bit parity problem with the Hessian. We begin exploring the concept of Hessian of loss functions of VQCs with a warp-up activity of solving a four-bit parity problem. Before illustrating the variational quantum classifier, the partial function is defined as a Boolean function whose output is 1 if and only if the input vector has an uneven number of ones. It is also called the XOR function of two inputs. The n-bit parity function is given as [50]

$$f : v \in \{0,1\}^{\otimes n} \rightarrow w \begin{Bmatrix} 1 \text{ if odd number of 1's in v} \\ 0 \text{ else} \end{Bmatrix} \tag{8}$$

The first step is to encode the input vectors into a quantum state. In a warm-up example, the inputs are 4-bit strings that are encoded into the state of qubits. The feature mapping and variational circuit of the four-bit parity problem
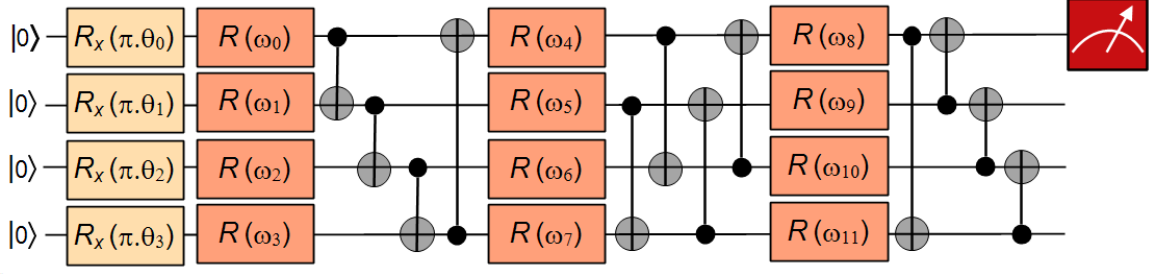
FIG. 2: **Feature map and variational quantum circuits for the parity problem.** A four-qubit feature map consisting rotation around x-axis to prepare the initial state. It is followed by three layers of variational gates, where each layer consists of rotational gates with three trainable parameters $(R(\omega_j) = R(\varphi_1, \varphi_2, \varphi_3))$ in each of the four qubits, followed by a set of CNOT gates. For two classes, the measurement is performed on one qubit, which is enough to have orthogonal measurements for the classes.

are shown in Fig 2. The initial layer of R_x gates prepares the initial state, which is also known as feature map circuit. Later, there are three layers of variational gates, where each layer consists of rotational gates with three trainable parameters in each of the four qubits, followed by a set of CNOT gates. It is to be noted that the combination of CNOT gates in each layer has a different structure. The variation circuit consists of 36 parameterized gates and 12 non-parameterized gates.

The input data $\overrightarrow{v}$ is encoded into quantum state using feature map function as $\psi : \overrightarrow{v} \rightarrow |\psi(\overrightarrow{v})\rangle \langle\psi(\overrightarrow{v})|$. Initially, one qubit rotations are performed around x-axis. It is followed by U_1 single-qubit gate to apply a quantum phase to the qubit. Furthermore, a controlled-NOT multi-qubit gate is applied to flip the target qubit when the control qubit is in $|1\rangle$. The purpose of the feature map circuit is to map the classical input data into the quantum state.

The final n-qubit feature quantum state becomes

$$|\psi(\overrightarrow{v})\rangle = U_{\psi(\overrightarrow{v})} |0\rangle^{\otimes n} \tag{9}$$

Second, a short depth quantum circuit $V(\theta)$ is applied to the feature state. It depends upon the selection of parameterization for the gates and the number of d layers. The classical optimizer handles the parameters during training to reduce the value of a loss function. Before returning a final classifier outcome, classical postprocessing is performed to the expectation value of the circuit. The aim is to determine the optimal classifying circuit $V(\theta)$ that separates the dataset with distinct labels. In variational quantum circuit, we used the $R_y(\theta)$ and $R_z(\theta)$ parameterized gates that are applied to rotate the qubits by angle (θ) around y-axis and z-axis, respectively. CNOT gates follow it. The objective is to find a sequence of gates that forms a final state $|\psi\rangle_0$. A cost function (C_f) is defined as the square of trace distance (D_t) between final $|\psi\rangle_0$ and initial state $|\psi\rangle = V(\theta)^\dagger |0\rangle$, which is determined as

$$C_f = \text{Tr}[O_f V(\theta) |\psi\rangle_0 \langle\psi|_0 V(\theta)^\dagger] \tag{10}$$

where $O_f = 1 - |0\rangle \langle 0|$. It is equivalent to $C_f = D_t(|\psi\rangle_0 \langle\psi|)^2$. As a warm-up to our study, the performance of a variational quantum classifier is tested for a parity problem. It is learnable in a quantum setting, a binary classification task $w = \{0, 1\}$ and its outcomes are measured on the computational basis. We selected the third qubit in the Pauli-Z direction to perform the measurement and thresholding (Δ) the expectation value $\langle Z \rangle \leq \Delta$ $(\langle Z \rangle > \Delta)$ to classify the input vectors in to one of the labels $w=0$ $(w=1)$, respectively. We utilized the gradient descent optimizer to iterate a parameter update based on the gradient of the loss function. We now consider the concept of Hessian that how it helps to realize the loss landscape (or curvature information) for the parity problem. Firstly, the parameters are initialized randomly and determine the set of parameters that can produce the target state. The optimization problem is translated into the loss function minimization.

It is presented as a function of θ_3 and θ_7, and rest of the parameters are set to the optimal values. Fig 3 shows the loss landscape of a parity problem with a local loss function of parameters, where $(\theta_1 = \theta_3$ and $\theta_2 = \theta_7)$. We started with some random initialization. The contour plots Fig 3 (b) and (d) show the direction of improvement in the optimization process that how the parameters $(\theta_3$ and $\theta_7)$ are optimized during 100 iterations. The green color point denotes the optimal value. The zoom-out and zoom-in versions of contour plots are shown in Figs (b) and (d) for a better view of optimal values.

The minimum value of a loss can be recovered with the gradient descent optimizer due to the point of local convexity in the landscape. Fig 4 depicts the behavior of eigenvalues of the Hessian for parity problem during training. Fig 4 (a-d) shows the variations between the minimum and maximum eigenvalues at each iteration. The distribution of
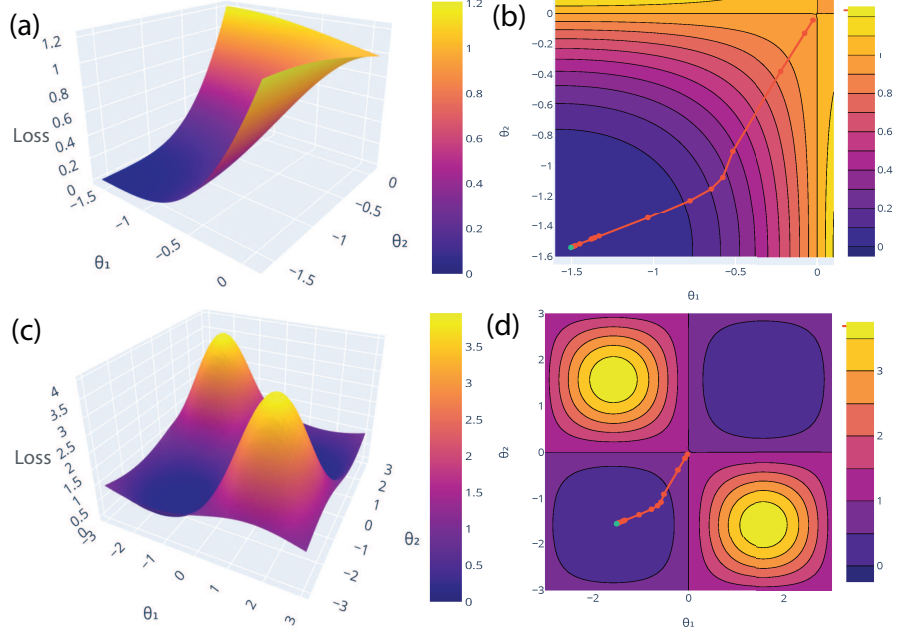
FIG. 3: **Loss landscape of the parity problem for** $(\theta_3$ **and** $\theta_7)$. In fig (a, c), the loss landscape is visualized with a local loss function for two parameters θ_3 and θ_7, where rest of the parameters are set to the optimal values. Moreover, it cannot be visualize for more than 2 parameters for the 3D Loss, because it cannot obtain the full range of loss between (0 and 1). In fig (b, d), the points in contour plots show the direction of improvement in the optimization process i.e. how the parameters are optimized during iterations. Fig (b) is just the zoom-in version of Fig (d) for better clarification. The green color point depicted the optimal value of θ_3 and θ_7.

eigenvalues for the randomly initialized quantum circuit shows the mixture of negative and non-negative values close to zero at iterations (0-7). Fig 4 (b-c) shows some of the eigenvalues are positive, some are negative, and the bulk of them is zero. Fig 4 (d) shows a loss for the well converged variational circuit where we left with a single negative eigenvalue of the Hessian and rest all are non-negative (at 100th iteration). In fact, the zero gradient ensures that it is a global minimum. Moreover, the zero eigenvalues correlate to directions where variations in parameters do not alter the curvature information. Thus, Hessian's behavior of the eigenvalues helps to obtain the minimum and maximum stability in the directions of loss landscape.

B. Classification of Diabetes

Let us consider the case of the classical diabetes dataset for the classification of diabetes in supervised learning. In this section, we analyzed the loss landscape through the Hessian for diabetes dataset and investigated how VQC will perform to predict diabetes. The diabetes dataset can be download from UCI machine learning repository [49]. It consists of 8 input features (age, glucose, insulin, pregnancies, body mass index (BMI), skin thickness, diabetes pedigree function, and blood pressure) and one binary output feature.

Firstly, the 8 input features are encoded into the state of qubits by a quantum feature map. Consider a classical dataset $D = \{(x^n, y^n)\}_{n=1}^{S}$ for binary classification, where $y^n \in \{0, 1\}$ i.e. 0 for no diabetes and 1 is for diabetes. Each segment of classical data is encoded into an amplitude of a qubit using single-qubit rotations. Afterward, a variational quantum circuit $V(\theta)$ is applied to the feature quantum state for training and classification of diabetes. The 8-qubit variational quantum circuit with a local cost function is constructed, as shown in Fig. 5. The feature map consists of Hadamard gates, rotation around y-axis and control-Z entangling gates. It is followed by a variational part of the circuit containing single-qubit rotations $R(\omega_j) = R(\varphi_1, \varphi_2, \varphi_3)$ on each qubit.

The feature map and variational quantum circuits are repeated two times. It is constructed simple enough so that it can be executed on real quantum systems and complex to separate the input data after mapping. For our experiments, the feature map and variational quantum circuits were constructed to a fixed number of depth and qubits. To evaluate the most likely state of an outcome qubit, the quantum circuit can be determined for several iterations by considering a similar input to calculate the probability distribution among the basis states. We used the gradient descent optimization method to find the parameters which can determine the probabilities closest to reality.
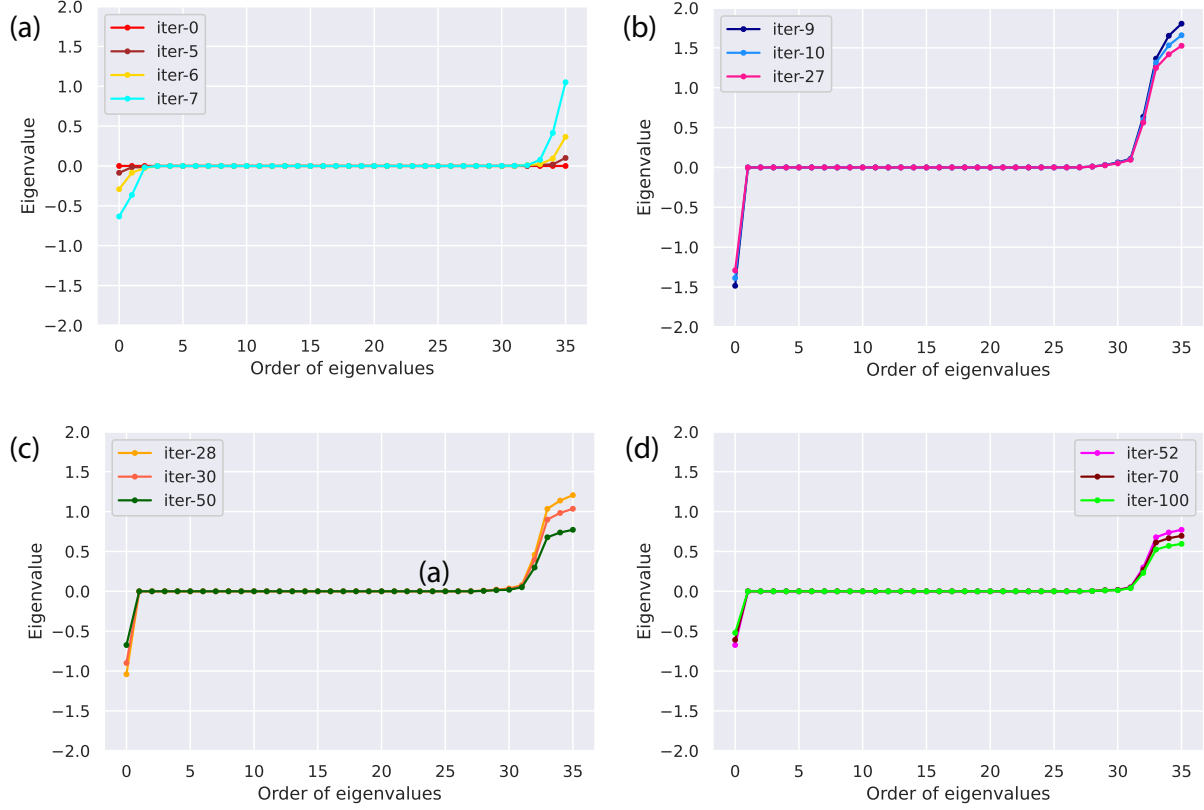
FIG. 4: **The evolution of behavior of the eigenvalues of the Hessian during training for the parity problem**. Initially, it shows the mixture of non-negative and negative values. Finally, a well converged loss is observed where the most of the Hessian's eigenvalues are non-negative. During the training, we separate the curves for better clear view as the difference between smallest and biggest eigenvalues is changing for different epochs.
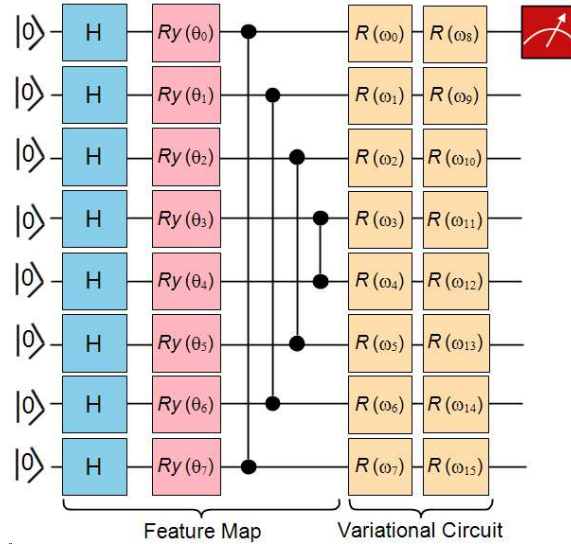

FIG. 5: **Variational quantum circuit**. A general 8-qubit variational quantum circuit is constructed for diabetes dataset with a local cost function. In feature map, Hadamard gates, rotation around y-axis and control-Z entangling gates are applied. The variational part of circuit consisting parameterized rotations $R(\omega_j) = R(\varphi_1, \varphi_2, \varphi_3)$ applied on each qubit.
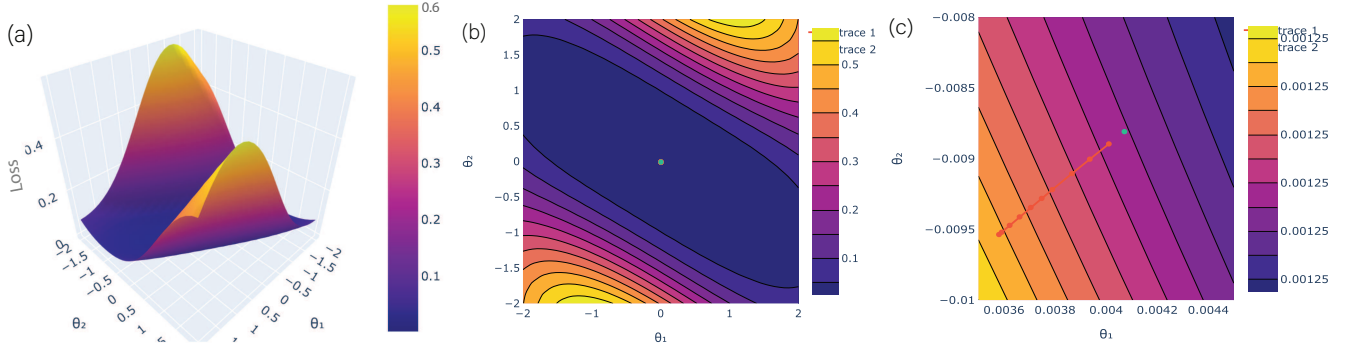
FIG. 6: **Loss landscape of the diabetes dataset for (θ_0 and θ_{24}).** (a) The loss landscape is demonstrated with a local loss function for two qubits θ_0 and θ_{24}. Fig (b-c) The direction of improvement in the optimization process is shown using contour plots i.e. how the parameters are optimized during iterations.
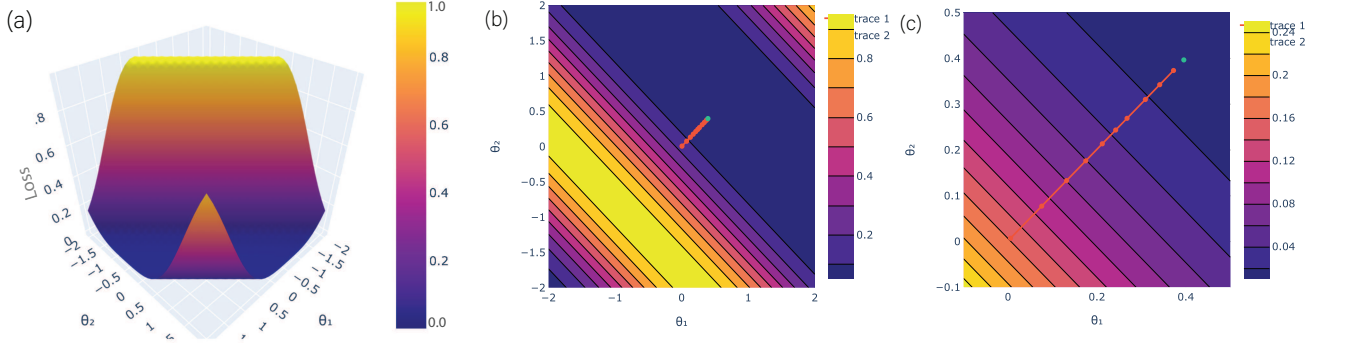

FIG. 7: **Loss landscape of the diabetes dataset for (θ_1 and θ_{25}).** (a) The loss landscape of the diabetes dataset is visualized with a local loss function for two parameters θ_1 and θ_{25}. Fig (b-c) give a clear view of the optimal values during iterations.

The measurement is performed on a qubit by employing the Pauli operators in a specific direction. The outcome qubit discovers the predicted value of an input, i.e., the class label values allocated. The loss landscape of VQC for a diabetes dataset with a loss function of parameters (θ_0) and (θ_{24}) is visualized in Fig 6, where θ_1 and θ_2 are set to θ_0 and θ_{24}, respectively. Here, the other parameters are set to the optimal value after each iteration. The input vectors are normalized to lie in $[-\pi, \pi]$ and plotted the loss landscape for two qubits in a range. If we try to plot the loss landscape for more than two parameters, then it cannot obtain the full range between (0 and 1). The contour plots (b) and (c) show the evolution of optimized parameters during 30 iterations in Fig 7. Furthermore, the loss landscape with a loss function of parameters (θ_1) and (θ_{25}) has been analyzed in Fig 7. It has been observed from the landscape that optimum cost is single with good local minima. It shows the prediction map of VQC with Z-measurement for the diabetes dataset.

The distribution of Hessian's eigenvalues is determined over the training process of a variational circuit to locate one of the minima. It is used to investigate whether a particular stationary point is a saddle point or not. At the beginning of the training, the gradient descent method is struggling to break the symmetry. Due to the small gradients, it faces a problem in training small quantum circuits. Fig 8 (a) shows a distribution of the Hessian's eigenvalues consisting of equally possible negative and positive values, and most of them are zero for iterations (0, 2, and 4). We observed that the negative eigenvalues gradually started to disappear with the increase in number of iterations, as shown in Fig 8 (c). After the convergence at 30th iteration, a single negative eigenvalue is left and rest all became non-negative, in Fig 8 (d). The bulk of zero eigenvalues shows a flat direction of the $f(\theta, \overrightarrow{x})$ in the loss landscape, where any alterations in circuit parameters do not disturb the loss landscape. The positive semi-definite behavior of the Hessian's eigenvalues signifies a very steady result. Although, it is not practical to visualize the loss landscape of VQC in 3-dimensional due to the problem of fixing the other parameters. Nevertheless, it is feasible to visualize the loss landscape of variational quantum classifiers through the lens of the Hessian.
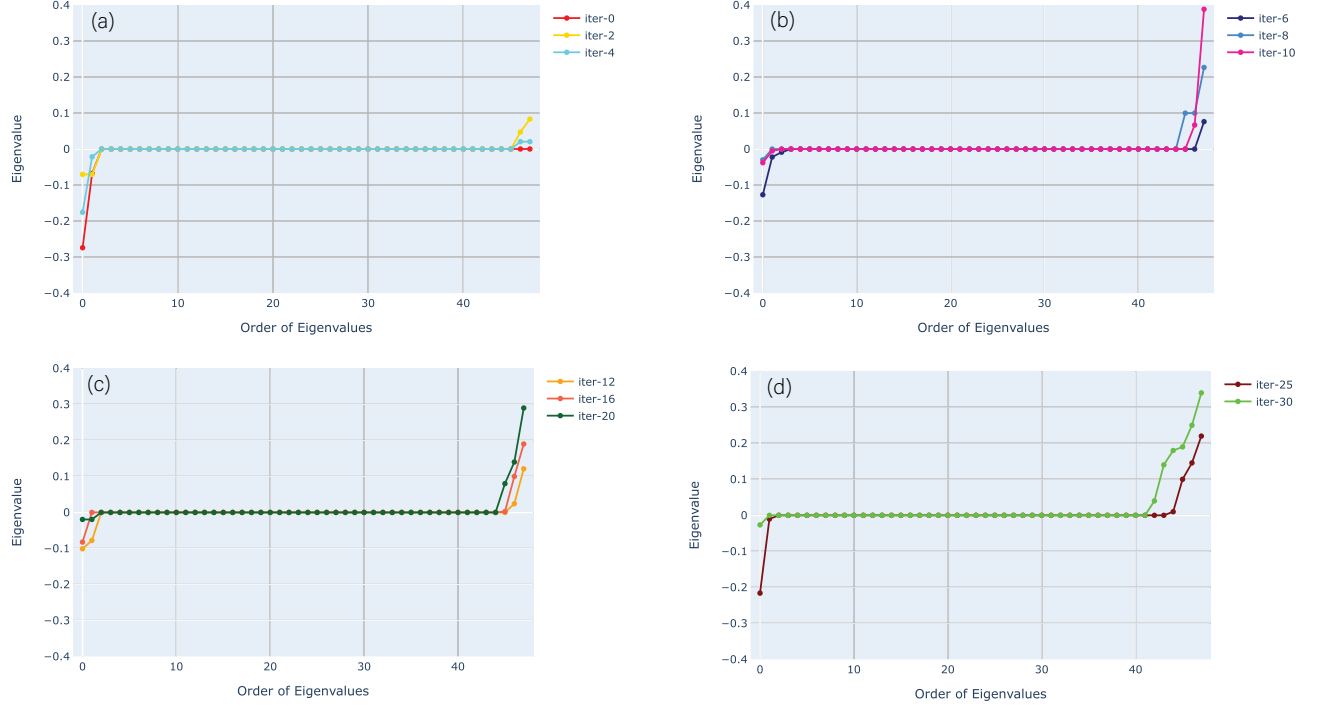
FIG. 8: **The evolution of behaviour of the eigenvalues of the Hessian during training for the diabetes dataset.** It shows the eigenvalues evolution during the training for different epochs. Initially, the Hessian's eigenvalues consisting of negative and positive values equally, and many of them are zero. After the convergence, it shows maximum positive eigenvalues.

IV. CONVERGENCE VIA ADAPTIVE HESSIAN LEARNING RATE

In this section, we show how the Hessian can adjust adaptively to the learning rate (LR) for each parameter. Adaptive learning rate has been a popular practice in classical machine learning. In any gradient based optimization method, a very large learning rate can cause an overshoot in cost while coming close to smaller gradients, whereas very small LR will cause an extremely slow approach towards the lowest point in loss landscape. Therefore, a smart trade off between both leads to the requirement of tuning the Learning rate properly. It is not as simple as it looks. We used an adaptive learning rate, which initially starts with the higher values of learning rate and gradually reduces it with reducing gradient values.

Adaptive Hessian learning rate (A-HLR) is something similar to the previous concept with an add on, specific to the quantum machine learning model's loss landscape analysis. Here, instead of using continuous variation of learning rates, we consider a set of discrete values of learning rates. We begin with the largest LR and gradient descent optimization process. After each step, we compare the updated cost with the previous cost. After few consecutive steps, if the difference between updated and previous cost is below a threshold level, then LR is set to the next smaller discrete value in the set of LR's. The complete process is repeated until we reach the lowest LR in the set of discrete learning rates. Since the loss landscape of quantum machine learning models is quite different from the classical machine learning models. There can be observed a specific repetitive manner of landscape unlike classical model, which leads to the higher probability of getting stuck in local minima.

In case, if the execution reached lowest LR among the set of discrete learning rates, there can be two possible cases (i) either the optimization stuck in a local minima, we need to get out of this, (ii) or the optimization problem reached the global minima. To tackle these scenarios, we considered the concept of Hessian matrix. The set of eigenvalues of Hessian matrix are calculated on the current parameter set. If the count of negative eigenvalues is lesser than a threshold value, then the solution obtained is a global minima. In case, if the count is greater, then it got stuck in local minima. It is to be noted that the threshold value of negative eigenvalue count is a hyperparameter, which has to be tuned properly depending on the type of data in a given dataset. (Practically it should be zero, but in reality its not always possible to reach that point). If the model is stuck at local minima, i.e. the learning rate has already reached the lowest of its all possible given values. Then, the optimization will be again started with the higher learning rate with an objective that the cost will overshoot and come out of the local minima. The process is repeated until the LR comes to its lowest value. Finally, it is evaluated whether the optimization has reached global minima or not using
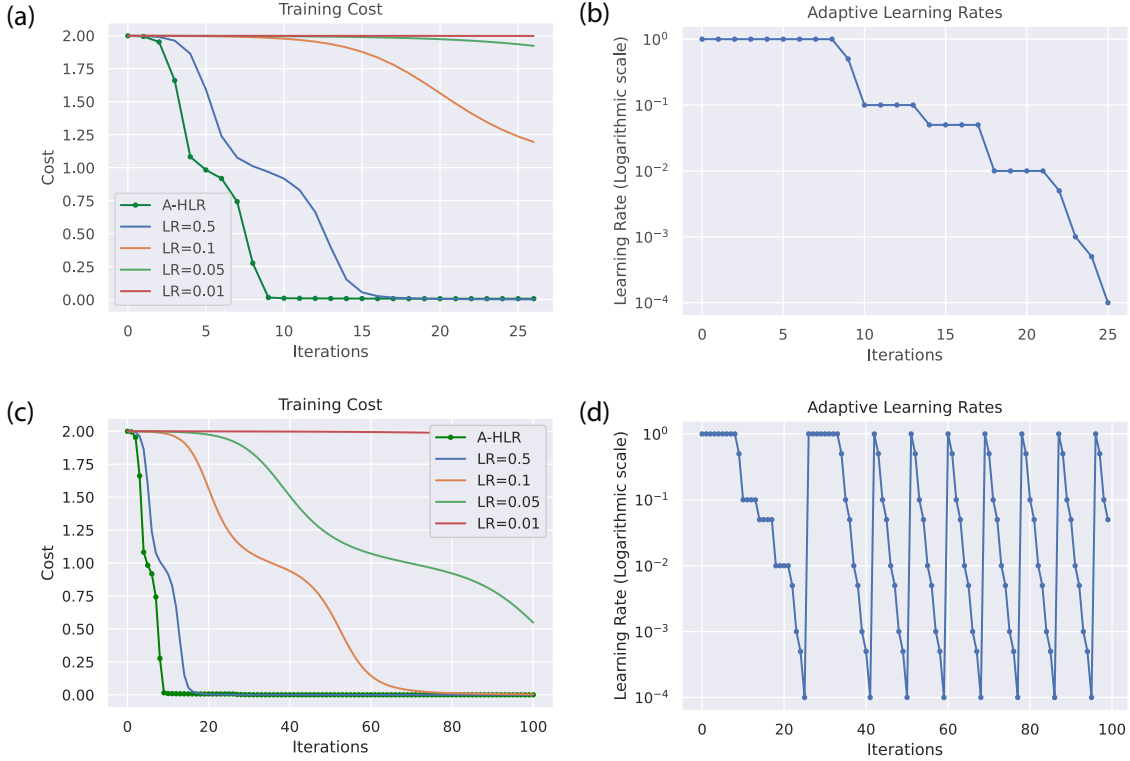
the Hessian matrix.

FIG. 9: **Training cost of four bit parity dataset with adaptive Hessian learning rate (A-HLR).** In fig (a, c), the comparison is shown between the adaptive Hessian learning rate and gradient descent method with constant learning rates. A-HLR shown faster convergence than gradient descent with fixed learning rates. It has shown the stable and efficient convergence of the cost function for parity dataset during training. Fig (b, d) shows how A-HLR evolves during training of 25 and 100 iterations. It depicts how to overshoot the local minima of the cost function using adaptive Hessian lower to higher learning rates.

Fig 9 (a) shows the comparison that how the cost is evolving throughout the optimization process with an adaptive Hessian learning rate (A-HLR) method and gradient descent method with constant learning rates. It has been determined that A-HLR converged very well within 25 iterations. The gradient descent method with LR=0.5 learning rate also converges, but not as quick as with A-HLR. Fig 9 (b) shows how the value of adaptive learning rate evolves on using A-HLR during first 25 iterations and depicts how it overshoots the local minima of the cost function using lowest to highest LRs. Fig 9 (c) shows the comparison between A-HLR and gradient descent methods with different learning rates for 100 iterations. Fig 9 (d) depicts how the value of adaptive learning rate evolves on using A-HLR during the first 100 iterations. A-HLR fits the local shape of gradient to the loss landscape very well, provides a faster convergence than gradient descent method with constant learning rates. It permits one to select a descent direction for faster convergence during the training of variational circuits. Therefore, the local traps in the loss landscape can be avoided by using the adaptive hessian learning rate approach.

V. CONCLUSION

In this paper, the curvature information of the loss landscape of variational quantum classifiers has been visualized via the lens of Hessian eigenvalues. We developed a simple theoretical quantum model of Hessians and gradients, as justified by datasets for numerical justifications on VQCs. The parity function problem is considered as a warm study to show the behavior of Hessian's eigenvalues. It has observed that VQC has an exceptional ability to generalize small datasets. Furthermore, we visualized the cost function landscape of VQC designed for the diabetes dataset. It converges efficiently for data-driven problems. We identified some differences in the convergence with adaptive Hessian learning rate and gradient descent method using fixed learning rate. It has been observed that adaptive Hessian learning rate helps to overshoot the cost if it gets fall into local minima and converge quickly. It seems

beneficial to study the local curvature information of VQC through the Hessian. The integration of gradient-based methods and Noisy Intermediate-Scale Quantum (NISQ) devices is still a young area and potentially has a lot more to offer. In the future, this work will open up new avenues of research in solving classical and quantum optimization problems and framework design. It will help the research communities to accelerate the analysis of variational quantum algorithms based on Hessian.

DATA AVAILABILITY

The data that support this study are available from the corresponding author upon reasonable request.

AUTHOR CONTRIBUTIONS

All authors have contributed equally to this work.

ADDITIONAL INFORMATION

Competing interests: The authors declare no competing interests.

---

[1] J. Preskill, Quantum computing in the nisq era and beyond, Quantum 2 (2018) 79.

[2] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, J. L. O'brien, A variational eigenvalue solver on a photonic quantum processor, Nature communications 5 (2014) 4213.

[3] O. Higgott, D. Wang, S. Brierley, Variational quantum computation of excited states, Quantum 3 (2019) 156.

[4] K. M. Nakanishi, K. Mitarai, K. Fujii, Subspace-search variational quantum eigensolver for excited states, Physical Review Research 1 (3) (2019) 033062.

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

[6] A. Garcia-Saez, J. Latorre, Addressing hard classical problems with adiabatically assisted variational quantum eigensolvers, arXiv preprint:1806.02287 (2018).

[7] R. M. Parrish, E. G. Hohenstein, P. L. McMahon, T. J. Martínez, Quantum computation of electronic transitions using a variational quantum eigensolver, Physical review letters 122 (23) (2019) 230401.

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

[9] M. E. Morales, J. Biamonte, Z. Zimborás, On the universality of the quantum approximate optimization algorithm, Quantum Information Processing 19 (9) (2020) 1–26.

[10] J. Stokes, J. Izaac, N. Killoran, G. Carleo, Quantum natural gradient, Quantum 4 (2020) 269.

[11] E. Farhi, H. Neven, Classification with quantum neural networks on near term processors, arXiv preprint:1802.06002 (2018).

[12] M. Altaisky, Quantum neural network, arXiv preprint quant-ph/0107012 (2001).

[13] M. Schuld, I. Sinayskiy, F. Petruccione, The quest for a quantum neural network, Quantum Information Processing 13 (11) (2014) 2567–2586.

[14] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, S. Woerner, The power of quantum neural networks, arXiv preprint:2011.00027 (2020).

[15] I. Cong, S. Choi, M. D. Lukin, Quantum convolutional neural networks, Nature Physics 15 (12) (2019) 1273–1278.

[16] J. Romero, J. P. Olson, A. Aspuru-Guzik, Quantum autoencoders for efficient compression of quantum data, Quantum Science and Technology 2 (4) (2017) 045001.

[17] A. Pepper, N. Tischler, G. J. Pryde, Experimental realization of a quantum autoencoder: The compression of qutrits via machine learning, Physical review letters 122 (6) (2019) 060501.

[18] J. Romero, A. Aspuru-Guzik, Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions, arXiv preprint:1901.00848 (2019).

[19] A. Uvarov, J. Biamonte, On barren plateaus and cost function locality in variational quantum algorithms, arXiv preprint:2011.10530 (2020).

[20] J. Kim, J. Kim, D. Rosa, Universal effectiveness of high-depth circuits in variational eigenproblems, arXiv preprint:2010.00157 (2020).

[21] E. Campos, A. Nasrallah, J. Biamonte, Abrupt transitions in variational quantum circuit training, arXiv preprint:2010.09720 (2020).

[22] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, H. Neven, Barren plateaus in quantum neural network training landscapes, Nature communications 9 (1) (2018) 1–6.

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

[24] E. Grant, L. Wossnig, M. Ostaszewski, M. Benedetti, An initialization strategy for addressing barren plateaus in parametrized quantum circuits, Quantum 3 (2019) 214.

[25] H. Li, Z. Xu, G. Taylor, C. Studer, T. Goldstein, Visualizing the loss landscape of neural nets, in: Advances in neural information processing systems, 2018, pp. 6389–6399.

[26] Y. Cooper, The loss landscape of overparameterized neural networks, arXiv preprint:1804.10200 (2018).

[27] G. M. Rotskoff, E. Vanden-Eijnden, Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error, stat 1050 (2018) 22.

[28] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, P. J. Coles, Absence of barren plateaus in quantum convolutional neural networks, arXiv preprint:2011.02966 (2020).

[29] K. Zhang, M.-H. Hsieh, L. Liu, D. Tao, Toward trainability of quantum neural networks, arXiv preprint:2011.06258 (2020).

[30] L. Sagun, U. Evci, V. U. Guney, Y. Dauphin, L. Bottou, Empirical analysis of the hessian of over-parametrized neural networks, arXiv preprint:1706.04454 (2017).

[31] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, P. T. P. Tang, On large-batch training for deep learning: Generalization gap and sharp minima, arXiv preprint:1609.04836 (2016).

[32] P. Rebentrost, M. Schuld, L. Wossnig, F. Petruccione, S. Lloyd, Quantum gradient descent and newton's method for constrained polynomial optimization, New Journal of Physics 21 (7) (2019) 073023.

[33] P. Huembeli and A. Dauphin, "Characterizing the loss landscape of variational quantum circuits," *Quantum Science and Technology*, vol. 6, no. 2, p. 025011, 2021.

[34] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, et al., Variational quantum algorithms, arXiv preprint:2012.09265 (2020).

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

[36] M. E. Morales, J. Biamonte, Z. Zimborás, On the universality of the quantum approximate optimization algorithm, Quantum Information Processing 19 (9) (2020) 1–26.

[37] W. C. Thacker, The role of the hessian matrix in fitting models to measurements, Journal of Geophysical Research: Oceans 94 (C5) (1989) 6177–6196.

[38] C. Bishop, Exact calculation of the hessian matrix for the multilayer perceptron (1992).

[39] A. Van Den Bos, Complex gradient and hessian, IEE Proceedings-Vision, Image and Signal Processing 141 (6) (1994) 380–382.

[40] A. L. Yuille, A. Rangarajan, The concave-convex procedure, Neural computation 15 (4) (2003) 915–936.

[41] K. Mitarai, M. Negoro, M. Kitagawa, K. Fujii, Quantum circuit learning, Physical Review A 98 (3) (2018) 032309.

[42] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, N. Killoran, Evaluating analytic gradients on quantum hardware, Physical Review A 99 (3) (2019) 032331.

[43] M. Cerezo, P. J. Coles, Impact of barren plateaus on the hessian and higher order derivatives, arXiv preprint:2008.07454 (2020).

[44] K. Sharma, M. Cerezo, L. Cincio, P. J. Coles, Trainability of dissipative perceptron-based quantum neural networks, arXiv preprint:2005.12458 (2020).

[45] A. S. Bhatia, M. K. Saggi, A. Kumar, and S. Jain, "Matrix product state–based quantum classifier," *Neural computation*, vol. 31, no. 7, pp. 1499–1517, 2019.

[46] A. S. Bhatia and M. K. Saggi, "Implementing entangled states on a quantum computer," *arXiv preprint:1811.09833*, 2018.

[47] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, in: Advances in neural information processing systems, 2019, pp. 8026–8037.

[48] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri, et al., Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv preprint:1811.04968 (2018).

[49] Pima Indians Diabetes Database, `https://www.kaggle.com/uciml/pima-indians-diabetes-database`, [Online; accessed 14-Jan-2021].

[50] M. Arslanov, D. Ashigaliev, E. Ismail, N-bit parity ordered neural networks, Neurocomputing 48 (1-4) (2002) 1053–1056.