IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 015004 https://doi.org/10.1088/2632-2153/acb12f

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
6 December 2022

ACCEPTED FOR PUBLICATION
6 January 2023

PUBLISHED
20 January 2023

PAPER

The impact of cost function globality and locality in hybrid quantum neural networks on NISQ devices

Muhammad Kashif * and **Saif Al-Kuwari**

Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar

* Author to whom any correspondence should be addressed.

E-mail: mkashif@hbku.edu.qa

Keywords: quantum machine learning, quantum neural networks, barren plateaus, qubit measurements, cost function

Abstract

Quantum neural networks (QNNs) are often challenged with the problem of flat cost function landscapes during training, known as barren plateaus (BP). A solution to potentially overcome the problem of the BP has recently been proposed by Cerezo *et al* In this solution, it is shown that, for an arbitrary deep quantum layer(s) in QNNs, a global cost function (all qubits measured in an n-qubit system) will always experience BP, whereas a local cost function (single qubit measured in an n-qubit system) can help to alleviate the problem of BP to a certain depth ($\mathcal{O}(\log(n))$). In this paper, we empirically analyze the locality and globality of the cost function in hybrid quantum neural networks. We consider two application scenarios namely, binary and multi-class classification, and show that for multiclass classification, the local cost function setting does not follow the claims of Cerezo *et al*; that is, the local cost function does not result in an extended quantum layer's depth. We also show that for multiclass classification, the overall performance in terms of accuracy for the global cost function setting is significantly higher than the local cost function setting. On the other hand, for binary classification, our results show that the local cost function setting follows the claims of Cerezo *et al*, and results in an extended depth of quantum layers. However, the global cost function setting still performs slightly better than the local cost function.

1. Introduction

One of the greatest scientific revolutions of the 21st century is the development of machine learning (ML) and neural networks (NNs). Unlike traditional approaches, where explicit programming is required to perform a specific task, in ML, the machine is trained to learn from data [1] to solve the problems on new unseen data [2]. Today, ML algorithms have become a favorable choice for almost every plausible task ranging from healthcare [3] to marketing [4] and fraud detection [5].

Over the last few decades, artificial NNs, in particular deep neural networks, have seen tremendous progress and are proven to be successful for a variety of applications, such as question answering [6], object detection [7], and social recommendation [8]. Despite their phenomenal success, training NNs faces some challenges, such as overfitting [9]; in fact, training NNs has recently been shown to be an NP-hard problem [10, 11]. These challenges motivated new efforts to look for alternative approaches to deep learning.

Recently, the advent of noisy intermediate scale quantum (NISQ) devices [12–15] facilitated the exploration of different (classical) applications for the post-quantum era. Among others, quantum machine learning (QML) has recently emerged as a potential application of quantum computation as a by-product of conceptual cross-fertilization of quantum computation and ML [16–18]. QML focuses on achieving the quantum advantage in ML applications by harnessing the fundamental properties of quantum mechanics, like superposition and entanglement [19–22]. Quantum advantage essentially entails more efficient problem-solving than the best existing classical computers. Consequently, many ML algorithms have been

studied in the quantum domain, such as quantum linear regression [23], balanced k-means clustering [24], and quantum support vector machines [25–27].

A number of deep learning approaches for quantum systems have also been proposed [28–32]. However, the majority of these proposals are not suitable for NISQ devices; and those which are, do not have a one-to-one correspondence with classical artificial NNs [33]. Consequently, there has been a growing interest in developing hybrid approach (a combination of classical and quantum processing) that is also compatible with NISQ devices [34]. In the context of quantum neural networks (QNNs), hybrid quantum neural networks (HQNNs) are frequently been used to analyze the much-anticipated quantum advantage in QNN applications. The term *hybrid* in HQNNs essentially entails the classical input preprocessing, classically post-processing of the QNN's output, and classical optimization routines. Since the HQNNs entirely replicate QNNs with only the addition of input preprocessing and output post-processing, therefore HQNN's analysis is directly applicable to QNNs and vice versa.

To this end, parameterized quantum circuits (PQCs) emerged as promising quantum versions of artificial neurons [18, 22, 35–38]. PQCs are classically optimizable quantum circuits and are robust against device errors [39]. PQCs, followed by data embedding, are often called QNNs. QNNs have been proposed for various applications, such as transfer learning [40], generative modeling [41–45] and classification [22, 39, 46–51]. Analogous to classical NNs, where classical parameters are optimized during training, in HQNNs, the quantum parameters need to be optimized too [18, 30, 35, 46, 52], which is not a straightforward task. Therefore, a significant amount of research has been focused on the development of the quantum landscape theory [53], which explores the properties of HQNNs (or QML in general) cost function landscapes. Recently, there have been some interesting results in QNN's training landscapes, such as the presence of barren plateaus (BP) [54, 55], the existence of sub-optimal local minima [56], noise effects in cost function landscapes [57–60] and a global or local definition of the cost function to somehow overcome the (BP) problem [55, 61].

In this paper, we empirically investigate the globality and locality of the cost function in relation to the occurrence of BP in HQNNs for real-world applications, typically binary and multi-class classification. The globality and locality of the cost function in HQNNs essentially mean all and single qubit measurements respectively, in the underlying PQC a.k.a hidden quantum layers[1].

1.1. Related work

The phenomenon of BP is rapidly becoming a pressing challenge in QNNs training, first demonstrated in [54]. The BP is the phenomenon when the gradients in the cost function landscapes become exponentially flat with system size and are not further optimizable to reach the global minimum. In the introductory study on BP occurrence during QNNs training [54], the authors showed that for randomly initialized PQCs of $\mathcal{O}(\text{poly}(n))$, the variance of partial derivatives of the parameters vanishes exponentially with the number of qubits. However, their results were based on the fact that the PQCs subject to training forms 2-design, i.e. the ability to sample all the unitaries in Hilbert space. The PQCs capable of forming a 2-design are usually considered to be more expressible and in practice, a sufficiently deep PQC can approximate a 2-design, e.g. hardware efficient ansatz of depth $\mathcal{O}(\text{poly}(n))$ [62, 63].

The flat cost function landscapes make QNNs *untrainable*, resulting in significant performance deterioration, which limits QNN's use in real-world applications. The *trainability* (parameters subject to training are being optimized after every epoch) of QNNs has become challenging due to the problem of BPs. To this end, there have been some efforts to study the root cause of BPs in QNNs, which eventually would lead to a potential solution to avoid or at least delay the occurrence of BPs. Some recent studies reported possible causes of BPs, such as the noise levels in high-depth quantum circuits [57, 64–67], the approach used to encode the data into PQCs [22], and the type of entanglement being used [68]. These root causes of BPs in QNNs provided important insights that may lead to solutions. Consequently, a number of solutions have been proposed, including careful parameter initialization strategies in random PQCs [48, 69]. Another study [70] proposing the layer-wise training of QNNs shows that by incrementally increasing the underlying PQC's depth, the generalization can be reduced by a significant amount, showing enhanced trainability of QNNs. The initial study on BPs in QNNs [54] proved the existence of BPs for PQCs whose unitary transformation forms a 2-design, i.e. the underlying PQCs exhibit the maximum possible expressibility, ultimately leading to a conclusion that the underlying PQCs should indeed be very expressible to get exposed to BPs. This conclusion linked the occurrence of BPs with underlying PQC's expressibility. To this end, there have been some recent works analyzing the PQC's expressibility with respect to gradient magnitudes [37, 55, 71].

[1] From this point onward we use PQC, ansatz, and quantum layers interchangeably.

In [71], the authors show that expressible PQCs are more susceptible to BP as compared to the PQCs having lesser expressibility. However, expressible PQCs are still a favorable choice due to their potential to provide solutions for a wide range of problems. These results connect the BPs with the underlying PQC's depth in QNNs. A recent study investigated the existence of BPs for shallow PQCs in comparison with deep PQCs in [55], where it has been shown that by making the phenomenon of BP dependent on the cost function, it can be extended to shallow circuits (see section 2.3). Their results lead to the conclusion that along with other sources of flat cost function landscape, the underlying ansatz's[2] expressibility and cost function's globality can also induce the so-called BPs. The authors reported in [55], that the global cost function (all qubit measurement in an n-qubit system), irrespective of PQC's depth, will experience BP, whereas the local cost function (single-qubit measurement in an n-qubit system) will have a delayed occurrence of BP and hence allows more expressibility. The cost function globality and locality from the BP aspect is separately discussed in section 2.3. While these results are relevant in standalone QNNs, it is not clear how all these phenomena affect the trainability of HQNNs, which are of significant importance in NISQ regime.

1.2. Contribution

In this paper, we empirically analyze how BP affects the performance of HQNNs with respect to cost function globality and locality. We consider two real-world applications, namely: binary classification and multi-class classification. In our evaluation, we consider a single hardware-efficient periodic ansatz structure and vary the number of measurement observables (single and all-qubit observables in an n-qubit system). We train the PQCs (with both global and local cost functions) for different widths (number of qubits) and depths (number of times the PQC is repeated). The training results are then analyzed to verify the existence of BPs in HQNNs. Furthermore, PQCs with both global and local cost functions are compared to investigate the impact of the cost function globality and locality during HQNNs training. Our results demonstrate that for multiclass classification, HQNNs with global cost function setting perform significantly better than HQNNs with local cost function setting. However, for binary classification, both global and local function settings achieve comparable performance.

1.3. Organization

The rest of the paper is organized as follows: section 2 provides an overview of QNNs and their general structure. Moreover, the details about the hybrid QNNs and cost function globality and locality are also presented in this section. The complete methodology and experimentation details are presented in section 3. The experimental setup, along with dataset specifications are discussed in section 4. Section 5 presents and discusses the results of both PQCs (global and local) for binary and multi-class classification. The performance of both the PQC structures is also compared in this section. Finally, section 6 concludes the paper.

2. Background

In this section, we provide a brief background on QNNs and HQNNs. We discuss the general architecture of QNNs and describe it in detail. We also provide a brief background on data encoding approaches as used in this work. Furthermore, we discuss how parameter gradients are calculated in QNNs. We then provide similar details for HQNNS. Finally, we briefly define what cost functions are and discuss their globality and locality in the context of QNNs.

2.1. QNNs

QNNs contain parameterized gate operations [47, 72]. A typical architecture of QNN is depicted in figure 1. QNNs typically work in four steps, namely; (a) qubit initialization (b) classical to quantum feature mapping, a.k.a data encoding (c) unitary evolution via a PQC, and (d) qubit measurements which eventually give the output of the final network. The qubit initialization step is the definition of the number of qubits and is often dependent on the input size and the adopted data encoding strategy. The embedding or feature mapping step is usually not trained and optimized during the whole learning process, although there have been proposals to make the state preparation routines trainable [73]. The mapped features are then passed to the following PQC where they are trained and optimized for a particular application [31, 52, 74]. The optimization is usually performed by defining and minimizing a cost function.

Feature mapping from classical to quantum ($x \longrightarrow E(x)$) is a crucial step while constructing QNNs and may significantly impact the overall learning process. The most common embedding techniques for encoding

[2] In PQC context, ansatz is a quantum subroutine consisting of a sequence of gates applied to specific wires.

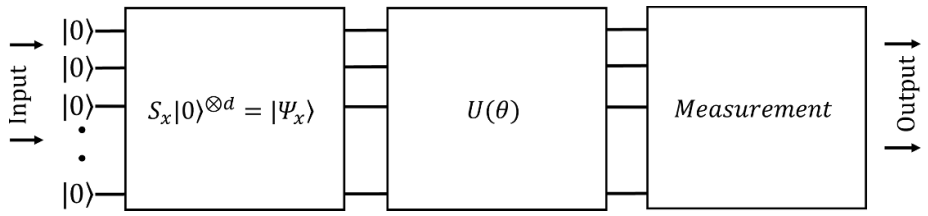


Figure 1. Typical QNN Architecture. The first box (from left) is the feature mapping step which prepares the quantum state $|\psi_x\rangle = |q_1, q_2, \ldots q_d\rangle$, which is then processed by PQC, i.e, $|\phi\rangle = \mathcal{U}(\theta)$ and finally the measurement is performed, i.e. $|f\rangle = \langle\phi|O|\phi\rangle$.

classical data into quantum states in QNNs are amplitude and angle embedding. Amplitude embedding [39] encodes the input into the qubit state vector. Equation (1) typically represents amplitude embedding, which can encode 2^n input features into an n-qubit system. A limitation of amplitude encoding is the requirement of normalized classical vectors leading to one less dimensional representation. Angle encoding [75], on the other hand, encodes the data into qubit rotation angles and is conjectured to be more suitable for QNNs [48, 76]. Equation (2) is a typical representation of angle encoding. Angle encoding usually encodes a single input feature per qubit and hence can encode n input features into an n-qubit system,

$$\begin{pmatrix} x_1 \\ \vdots \\ x_{2^n} \end{pmatrix} \leftrightarrow |\psi_x\rangle = \sum_{j=1}^{2^n} x_j |j\rangle \tag{1}$$

$$S_{x_j} = \bigotimes_{i=1}^{N} U_i \quad \text{where} \quad U_i := \begin{bmatrix} \cos\left(x_j^{(i)}\right) & -\sin\left(x_j^{(i)}\right) \\ \sin\left(x_j^{(i)}\right) & \cos\left(x_j^{(i)}\right) \end{bmatrix}. \tag{2}$$

Optimization is the ultimate goal in all ML tasks. In traditional ML, optimization is done by defining some cost function. The value of the cost function is then processed by an optimization algorithm, like gradient descent, which determines the cost minimizing direction by taking the derivative with respect to each parameter. In PQCs, the output, i.e. expectation value of observable(s), can be written as a *quantum function* ($f(\theta)$), which is parameterized by $\theta = \theta_1, \theta_2, \ldots$. The partial derivatives of ($f(\theta)$) can be expressed as a linear combination of other quantum functions. This is typically done by the same PQC, with only a difference in shift argument, leading to the fact that the same PQC architecture can be used to calculate the partial derivatives. The calculation of the partial derivative of a PQC using its parameter shifted instances is called the parameter shift rule [77, 78], and is shown in the following equation, where s is the shift argument,

$$\frac{df}{d\theta_i} = \frac{f(\theta_i + s) - f(\theta_i - s)}{2}. \tag{3}$$

2.2. HQNNs

Due to the limitations imposed by NISQ devices (mainly in the number of qubits), HQNNs are frequently used to analyze the quantum advantage of QNNs. HQNNs perform a particular learning task by exploiting both classical and quantum devices for training and optimization. In a typical HQNN architecture, a general QNN architecture (figure 1) is entirely replicated, as shown in figure 2. The only difference is the input preprocessing before passing it to QNN and classically postprocessing the output of QNN and interpreting it in a meaningful way.

The input preprocessing in HQNNs is generally performed to downscale the input size. This preprocessing step is important to cope with the limited number of qubits available in NISQ devices. The downscaling of input is usually performed through some dimensionality reduction algorithm or a classical layer with fewer neurons. It is important to note here that the input preprocessing step is not always needed and is dependent on the size of the input, the strength of the target quantum device, and the type of data encoding strategy used. Once the input data is within a reasonable limit, it is then passed to QNN to be mapped to quantum feature space and trained using a PQC, as discussed in section 2.1. Finally, the output of QNN is post-processed classically to obtain the final prediction/output. The postprocessing is usually done by a classical neuron layer, where a classical optimization routine is also defined. The non-linearity between the hidden layers in classical NNs is one of the major factors behind their tremendous success. On the other hand, the quantum operations are naturally linear and inducing non-linearity between quantum layers is

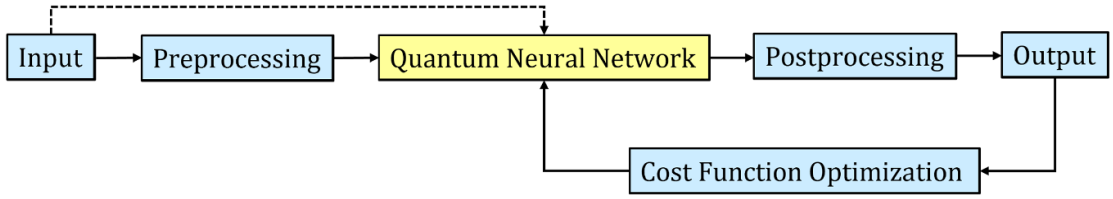


Figure 2. General Architecture of HQNN. The blue and yellow shaded regions are classical and quantum parts respectively. The dotted line from input to QNN represents the case when no input preprocessing is required and the data is directly passed to QNN.

challenging. This is where the HQNNs provide an added advantage along with input downscaling. The classical neurons layer at the end of QNN allows the application of a familiar non-linear activation from traditional ML.

2.3. Cost function

In ML, the cost function is usually used to evaluate the model's performance. The cost function typically measures how wrong the model is in finding a relation between the input and output. In other words, the cost function tells us how badly the model is performing. The objective of an ML model is to determine the optimal set of values for model parameters by finding the global minima of the cost function. This is typically done via an optimization algorithm. For classification problems, *cross-entropy* is a very popular and frequently used cost function. Similarly, gradient-based optimization algorithms such as *Adam optimizer* are commonly used. The optimizer iteratively updates the parameters by calculating the partial derivatives of the cost function with respect to model parameters until the best set of parameters (global minima) is reached. In PQCs, the output of PQCs can be differentiated using the parameter shift rule [77, 78].

In [54], where the issue of BPs in QNNs has first been discussed, the authors proved that for hardware-efficient ansatz, the BPs occur for the ansatz depth of order, $\mathcal{O}(\text{poly}(n))$. However, in a recently proposed solution to tackle BPs in QNNs [55], the authors claimed that the issue of BPs in relation to the underlying ansatz expressibility could be alleviated by making it dependent on the cost function. Consequently, two cost functions have been studied in QNNs [55], namely: *global* cost functions and *local* cost functions. These functions are described in the following equations:

$$C_{\mathrm{G}} = Tr\left[O_{\mathrm{G}} V(\theta)\,|\psi_0\rangle\langle\psi_0|\,V(\theta)^{\dagger}\right] \quad \text{with} \quad O_{\mathrm{G}} = 1 - |0\rangle\langle 0| \tag{4}$$

$$C_{\mathrm{L}} = Tr\left[O_{\mathrm{L}} V(\theta)\,|0\rangle\langle 0|\,V(\theta)^{\dagger}\right] \quad \text{with} \quad O_{\mathrm{L}} = 1 - \frac{1}{n}\sum_{j=1}^{n} |0\rangle\langle 0|_j \otimes 1_j \tag{5}$$

where 1_j represents the identity operation on all qubits except j.

Equation (3) refers to global cost functions, while equation (4) refers to local cost functions respectively. It is important to note here that both these equation represents the global and local cost function setting for an example of state preparation, where the goal is to find a gate sequence that prepares a target state $|\psi_0\rangle$. The observable O acts on all qubits in the global cost function setting, and it acts on a single qubit in the local cost function setting.

Furthermore, in [55], the authors assume that the ansatz subject to training forms a 2-design. Their main results show that for a global cost function, irrespective of the ansatz depth $(\mathcal{O}(1), \mathcal{O}(\log(n)), \mathcal{O}(\text{poly}(\log(n))), \mathcal{O}(\text{poly}(n)))$, the problem of BPs will always occur. However, in the case of the local cost function, the gradient's variance vanishes at worst polynomially, and the QNN is, therefore, trainable up to a depth of order $(\mathcal{O}(\log(n)))$. The BPs start appearing for depth of order $(\mathcal{O}(\text{poly}(n)))$ for local cost function and in between these regions, there is a transition region where gradients decay from polynomial to exponential.

3. Methodology

In this paper, we empirically analyze the recently proposed solution to somehow tackle the problem of BPs for shallow ansatzes in QNNs [55]. The solution aims to potentially delay the occurrence of BP, which enables an extended ansatz depth (more expressibility), resulting in a greater number of trainable parameters, and eventually leading to better performance. Although the proposed solution in [55] considers

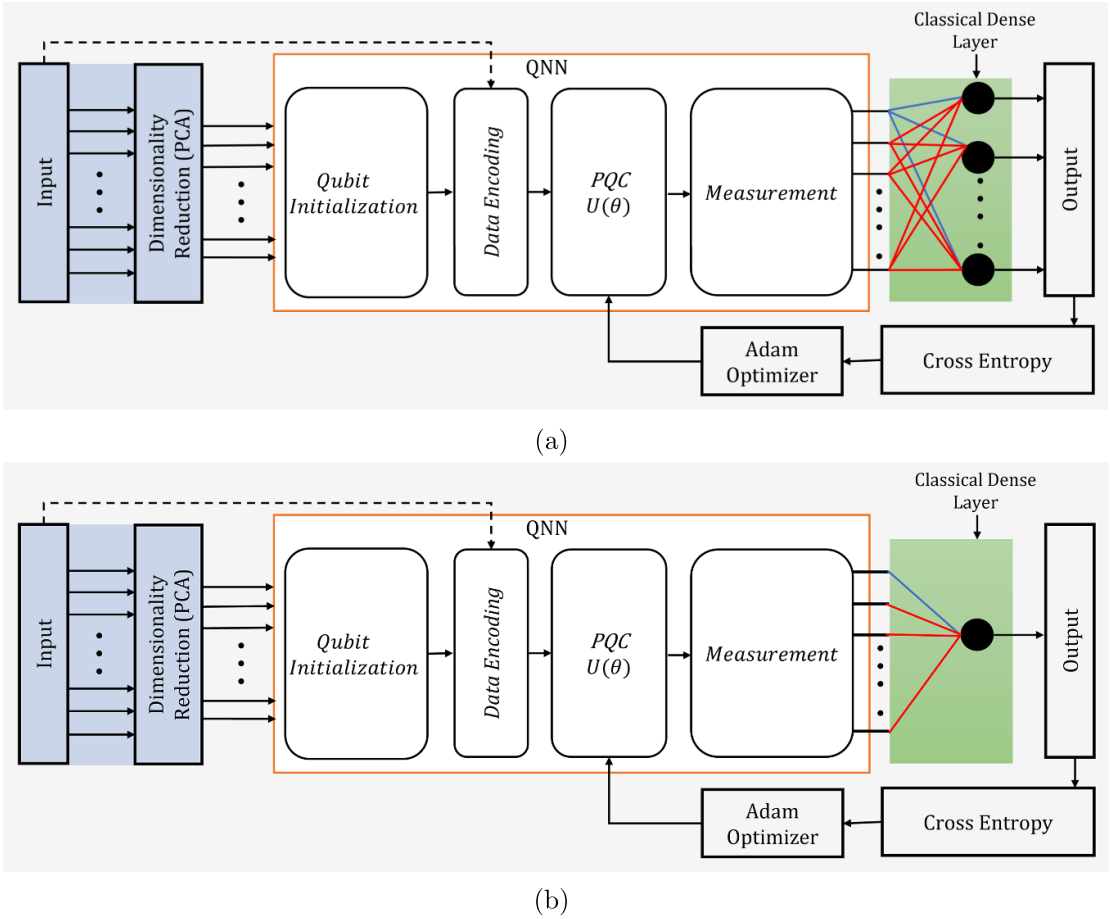


Figure 3. Complete HQNN architecture used in this work. (a) HQNN architecture for multiclass classification. The input is passed to the dimensionality reduction algorithm in the case of amplitude encoding, whereas the data is directly passed to QNN when data encoding is done via angle encoding. The blue lines are connected in the case of the local cost function and both blue and red lines are connected in the case of the global cost function. The blue and green shaded region denotes the input preprocessing and output post-processing respectively. (b) HQNN architecture update for binary classification. The classical dense layer in (a) is replaced with a single neuron. The red and blue lines are the same as in (a).

the standalone implementation of QNNs, we analyze its relevance in HQNNs for real-world applications. We typically consider two application scenarios for our analysis, namely, binary and multi-class classification. The complete architecture of HQNN used for our analysis is shown in figure 3.

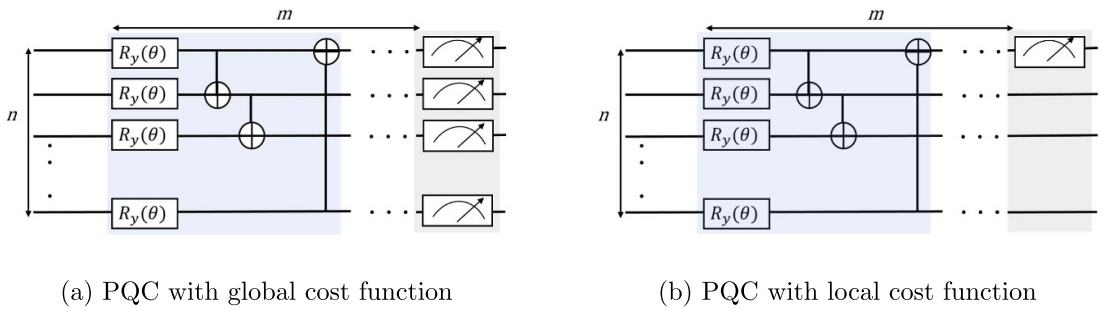
The methods in [55] apply to our HQNN design since the HQNNs used in this paper completely replicate the general QNN architecture from figure 1, with two additions: (a) input preprocessing to cope with the limitations of NISQ devices (mainly in the number of qubits), and (b) classical post-processing, to produce the final (classical) prediction of the network.

3.1. Components of HQNNs

We now discuss the three main components of our HQNN: input preprocessing (blue shaded region in figure 3), QNN, and output postprocessing (green shaded region in figure 3).

3.1.1. Input preprocessing

The input preprocessing step in HQNNs is typically performed because of the limited number of qubits available in NISQ devices. As discussed in section 2.2, the input dimensionality can be reduced by applying some dimensionality reduction algorithm or a classical layer with fewer neurons. Furthermore, the application of dimensionality reduction is dependent on the input feature size (to encode) and the type of data encoding strategy. We also reduce the input feature dimension of our input data to cope with the limitations of NISQ devices for one of the encoding schemes used in this paper, which we discuss in section 4.1, where we discuss the details of datasets used for both the target applications considered in this paper.



(a) PQC with global cost function (b) PQC with local cost function

Figure 4. Parameterized quantum circuits used in HQNN architecture. n denotes the width of quantum layers or in other words the number of qubits used. On the other hand, m denotes the number of times the ansatz (blue shaded region) is repeated before the measurement (light gray shaded region).

3.1.2. QNN construction

After the input preprocessing step, the next step is the construction of QNN. The QNN construction is a four-step process, as discussed in section 2.1. The first step is qubit initialization, which defines the total number of qubits and their initial state. The number of qubits (to define and initialize) is dependent on the size of input features and the type of data encoding. Furthermore, the qubit(s) can be initialized in any random state. However, in QNNs context, the qubits are usually initialized in the ground state, i.e. $|\psi\rangle^{\otimes n} = |0\rangle^{\otimes n}$, where n is the total number of qubits. We also initialize the qubits in the ground state for all the experiments performed.

The second step in HQNN construction is data encoding which encodes the classical input data to quantum states so that a quantum device can process it. We use two of the frequently used encoding techniques in HQNNs, namely angle (in *RY* rotations) and amplitude encoding.

The third step in QNN construction is the development of PQC where the encoded features are trained and optimized. The PQC contains single-qubit parameterized gates, which are (trainable) parameter-dependent gates, and multi-qubit gates for qubit entanglement. For the construction of PQC in our HQNN design, we use a single hardware-efficient periodic ansatz structure, first with global cost function (as shown in figure 4(a)) which we name as *global PQC* and then with local cost function (as shown in figure 4(b)) which we name as *local PQC*. We typically use single parameterized unitary ($R_y(\theta)$) per qubit and nearest neighbor entanglement[3] (because of the limitations of NISQ devices) using *CNOT* gate.

The last step in the QNN construction is qubit measurement, which is also a part of PQC construction. A qubit can be measured on an arbitrary basis, but the computational basis (also known as the eigenbasis of σ^z) is commonly used in QNNs. We also measure the qubits in σ^z basis in all the experiments to get the output of QNN.

3.1.3. Classical postprocessing

The last step in the HQNN construction is to post-process the measurement results of the enclosed QNN since the qubit measurement collapses the quantum state to a classical value that can be processed classically. We use a classical dense neuron layer, where all the neurons are connected to all the measured qubits in QNN, which is a single qubit in the case of the local cost function (blue lines in figure 3) and all the qubits in the case of global function (both blue and red lines in figure 3). The cost function is then defined based on which the optimization routine iteratively optimizes the parameters until the solution is reached.

3.2. Size of HQNNs

The total number of trainable parameters in a network governs the size of the network and can help in a fair inter-models comparison. For the HQNNs used in this paper, we generalize the following two equations to determine the total number of trainable parameters.

$$P_{\mathrm{Q}} = n \times m \tag{6}$$

$$P_{\mathrm{C}} = N + (N \times n_m) \tag{7}$$

[3] Nearest neighbor entanglement implies that neighboring qubits are entangled, where the first qubit is considered a neighbor to the last qubit.

where P_Q is the total quantum trainable parameters in the underlying network and P_C is the total classical trainable parameters. In equation (6), n and m denote the width and depth of the quantum circuit/layers, respectively. N and n_m in equation (7) represent the number of neurons (classical) in the last layer and the number of measured qubits, respectively. N depends on the total number of classes in the dataset. Therefore, in this paper, $N = 10$ for multi and $N = 2$ for binary classification.

4. Experimental setup

In this section, we discuss the experiments we performed, including the datasets we used for training the HQNNs for both target applications (binary and multi-class classifications) and the classical parameters specifications of HQNNs.

4.1. Dataset specifications

4.1.1. Multiclass classification

The *sklearn* digit dataset (handwritten digit images) is used to train the HQNNs [79]. The reason for choosing this particular dataset over some similar yet more popular datasets, such as MNIST, is because of the smaller input feature size, which is more suitable for NISQ devices. The sklearn's digit dataset contains a total of approximately 1797 samples, with a total of 10 classes and 180 samples per class. The training and testing data is divided into 75% and 25% respectively. Each data point is an image of 8×8 which results in an input feature dimension of 64.

As discussed in section 2.1, angle encoding encodes n features into an n-qubit system. Consequently, to encode the features with a dimension of 64, it would need 64 qubits, which is a relatively large number for NISQ devices. Therefore, we use an input dimensionality reduction algorithm, namely, principal component analysis (PCA), whenever we use angle encoding in this paper. The input dimension is reduced to a reasonable range, details of which are presented later in figure 5(a), where n typically denotes the width of PQC and so is the input feature dimension in case of angle encoding. The amplitude encoding on the other hand, encodes 2^n features into an n-qubit system, which for a feature dimension of 64 needs at least six qubits, which is a reasonable limit for NISQ devices. Therefore, in the case of amplitude encoding, the input feature dimensionality is not reduced, and the input is directly encoded.

4.1.2. Binary classification

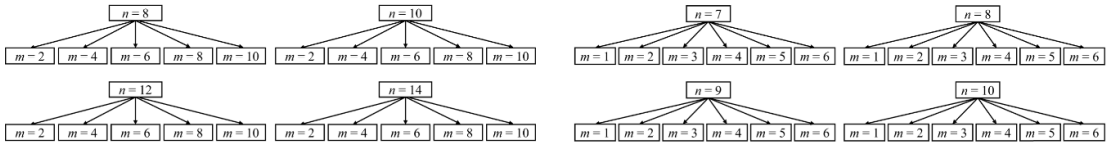
For binary classification, sklearn's breast cancer dataset [79] is used to train the HQNNs. This dataset contains two classes with a total of 569 samples, each data point with a dimensionality of 30. The malignant (positive) class has a total of 212 samples, and the benign (negative) class has a total of 357 samples. For the binary classification task, we only use amplitude encoding; we do not use PCA for input dimensionality reduction for the same reasoning as in the multiclass classification task. The reason why we used amplitude encoding only is discussed when we discuss the results of binary classification (see section 5.3).

4.2. Parameters specifications

The *categorical cross entropy* is used as a cost function in multiclass, whereas *binary cross entropy* is used as a cost function for binary classification. It is important to note here that the global and local loss functions shown in equations (3) and (4) are not combined with the above-mentioned loss functions for binary and multi-class classification. In fact, the loss functions in equations (3) and (4) are provided as a reference for global and local cost function setting, respectively, in QNNs. For the cost function optimization, we used *Adam* optimizer for both target applications (binary and multi-class classifications), as shown in figure 3. The initial learning rate for the optimization routine is set to 0.01. For better training, we use learning rate scheduling, which monitors the *validation loss*, and if there is no improvement in validation loss for three consecutive training iterations, it reduces the learning rate (new learning rate = previous learning rate $\times$ 0.1). Furthermore, a regularization technique named *early stopping* is used to avoid overfitting and stop the training if the validation loss is not reduced for four consecutive epochs. The batch size for the input data is 16 for multiclass classification and 8 for binary classification for all the experiments. The bigger and smaller batch sizes are in correspondence with the bigger and smaller size of datasets used. We used Pennylane [34] for constructing and training the HQNNs.

4.3. Experiments list

To make our analysis comprehensive, we perform an extensive list of experiments for both binary and multi-class classification problems, as shown in figures 5(a) and (b) respectively. The factor n denotes the width of quantum layers and is also the factor by which the feature dimension is reduced in the case of angle encoding. The reason for selecting a slightly bigger n in multi-class classification is because of greater input



(a) List of experiments for multiclass classification

(b) List of experiments for binary classification

Figure 5. The experiments performed in this paper.

dimensions. Similarly, the m, which denotes the depth of quantum layers, is smaller in binary classification because the dataset used for binary classification is simpler than that used in multi-class classification.

5. Results and discussion

In this section, we analyze the performance of both (global and local) PQCs from the aspect of BP for binary and multi-class classification. To analyze the BP in HQNNs for both global and local cost functions, we experiment with different widths (n) and depths (m) of the underlying PQCs. We benchmark the mean accuracy of HQNN for our analysis. We start by demonstrating the BP in a multiclass classification task for both the global and local PQCs individually with both the encodings used in this paper. We then perform the same analysis for binary classification tasks. We also analyze how the width and depth of quantum layers relate to the occurrence of BP in both cost function settings. We observe the significant performance difference with both the cost function settings, and we, therefore, compare HQNNs with both the cost functions for both the encoding used in this paper to better understand the role of cost function globality and locality in HQNNs.

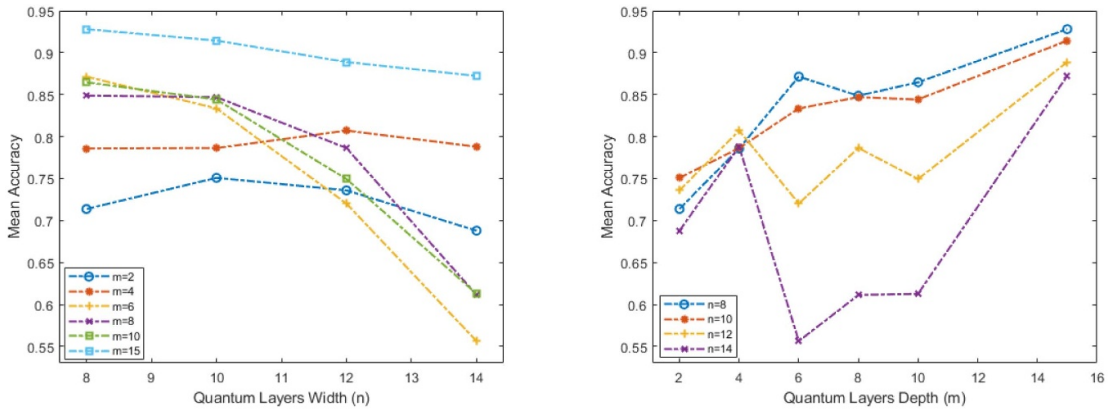
5.1. Multiclass classification

In this section, we analyze the existence of BP in a multiclass classification problem. We typically analyze the BP's existence in HQNNs with both global and local cost functions and with both the encodings used in this paper.

5.1.1. Global PQC with amplitude encoding

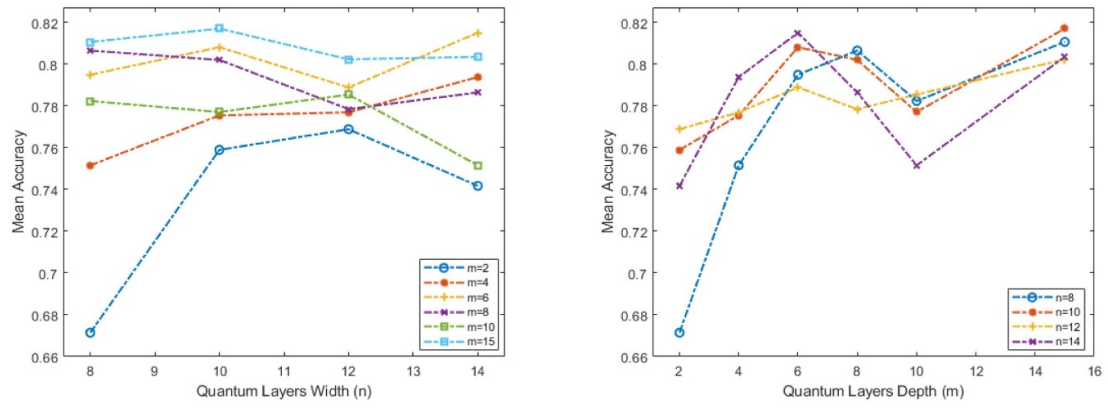
We first analyze the existence and potential effect of BP on the performance of HQNNs, when the cost function is global and the data is encoded in qubit state vectors. As discussed in section 4, the input feature size for the dataset used for multiclass classification is 64. To encode via amplitude encoding, a minimum of six qubits are required due to the fact that the amplitude encoding can encode 2^n features in n qubits, as discussed in section 2.1. The number of qubits (6) in this case is reasonable, given the limitations of NISQ devices. Therefore, we do not apply input dimensionality reduction (PCA) on the dataset used for multiclass classification in the case of amplitude encoding. We experiment with quantum layers of different widths n, and depths m, as shown in figure 5(a). The mean accuracy, both as a function of n and m, for global PQC and amplitude encoding, is presented in figure 6.

Analyzing the mean accuracy as a function of the number of qubits (n), there is a clear performance decline as n increases (figure 6(a)), exhibiting the presence of BP, which hinders the learning process of HQNNs and consequently degrades the accuracy. For almost all the depths (m), the accuracy deteriorates (or a negligible improvement) as the width n of underlying quantum layers is increased. Analyzing the mean accuracy as a function of m (figure 6(b)), we observe that for smaller widths ($n = 8$ and 10), the *allowable depth* of quantum layers is greater. The term *allowable depth* essentially means the performance improves as m increases. However, when n is increased ($n = 12$ and 14), the underlying quantum layers' depth tends to reduce significantly, and smaller m achieves better accuracy than relatively larger m. Although in bigger widths ($n = 12$ and 14), *very deep* quantum layers ($m = 15$)significantly improves the accuracy due to overparameterization (more training parameters than data points), it is still lower than smaller widths ($n = 8$ and 10), as shown in figure 6(b). Hence, we can safely say that increasing the number of qubits in HQNNs does not directly result in the occurrence of BP but is also dependent on the hidden quantum layer's depth. A *trade-off* between n and m is observed. This essentially means that to achieve comparable performance in HQNNs, the wider quantum layers would require relatively less depth and vice versa. Based on the results in figures 6(a) and (b), we conclude that deeper quantum layers with relatively shallow width result in slightly better performance.



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

Figure 6. Multiclass classification-mean accuracy of global PQC with amplitude encoding.



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

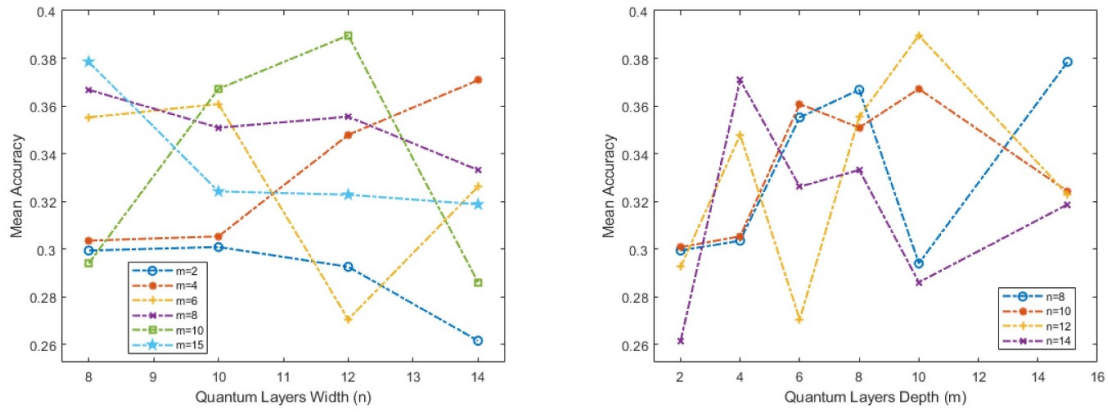
Figure 7. Multiclass classification-mean accuracy of Global PQC with angle encoding.

5.1.2. Global PQC with angle encoding

We now analyze the existence of BP in HQNNs for the multiclass classification problem with a global cost function setting (global PQC) and a different underlying encoding strategy, namely angle encoding. For angle encoding, as discussed in 2.1, an n-qubit quantum system is required to encode n features. As discussed in section 4, the feature size of the dataset used for the multiclass classification in this paper is 64. To successfully encode the data, a total of 64 qubits are required. Such a big number of qubits is not suitable for the NISQ era. Therefore, the input features dimension needs to be reduced. We apply PCA to reduce the input feature size to $8, 10, 12$ and 14, which are then encoded and passed to the following PQC for training. The number of training experiments for different n and m is then performed, as shown in figure 5(a). The mean accuracy for all experiments, both as a function of n and m is shown in figure 7.

In traditional classical ML, increasing the input training data with a fairly complex underlying network usually yields better performance. However, in HQNNs (with angle encoding), increasing the input data increases the quantum layer's width (number of qubits), which makes the underlying HQNN more prone to get exposed to BP, resulting in a hindered learning and eventually leading to performance decline.

As shown in figure 6(a), instead of an (expected) increasing accuracy trend with every increase in n for a fixed depth m, the accuracy starts declining after a certain n indicating the presence of BP. For relatively small (moderate) depth quantum layers ($m = 4$ and 6), there is a slight improvement in accuracy as n increases because of the *trade-off* between n and m, i.e. the smaller depth allows to have a relatively bigger width and vice versa. Similarly, when the depth m of underlying quantum layers increases, the performance starts declining after a certain m, for any fixed n, as shown in figure 6(b). Analogous to the case of amplitude encoding and global in figure 6, for a *very deep* quantum layer, the accuracy improves in general due to the so-called overparameterization phenomenon. However, even with overparameterization in play ($m = 15$),



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

Figure 8. Multiclass classification-mean accuracy of local PQC with amplitude encoding.

the smaller width (typically $n = 8, 10$) quantum layers yields better accuracy than wider quantum layers (typically $n = 14$). Based on the results in figure 7, we concur that similar to that of amplitude encoding, the deeper quantum layers with relatively shallow width provide better overall performance when encoding the input features in qubit rotation angles.

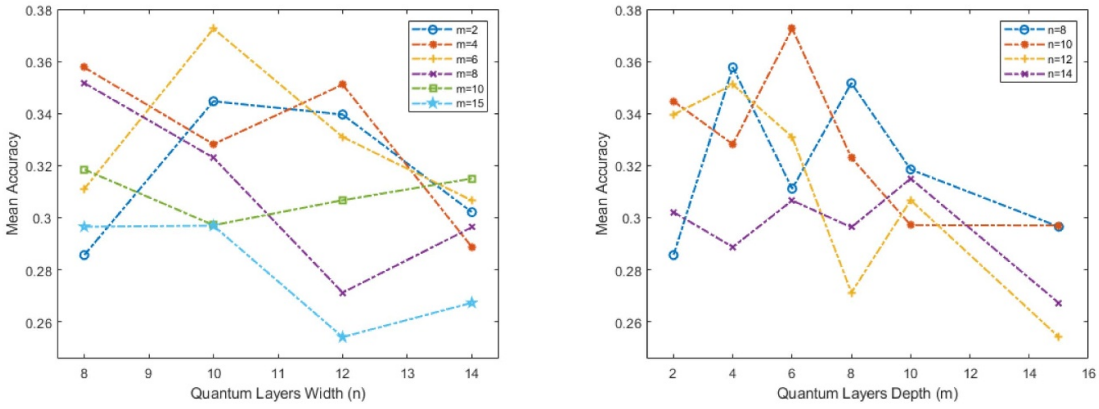
5.1.3. Local PQC with amplitude encoding

In this section, we analyze the HQNNs from the aspect of BP in a local cost function setting (local PQC), with the input data being encoded in qubit amplitudes. As per the claim in [55], the expectation is to have an increased accuracy trend as we increase the depth m quantum layers for a certain width n. Moreover, for bigger n quantum layers, the expectation is to have a greater allowable depth (m) of hidden quantum layers, eventually resulting in more trainable parameters and, consequently better learning. We experiment with the same n and m as in the case of global PQC and for both amplitude and angle encoding. Similar to the case of BP analysis of HQNNs with amplitude encoding for global cost function (section 5.1.1), the input is directly encoded with any dimensionality reduction. All the experiments from figure 5(a) are performed and the mean accuracy, both as a function of n and m, is presented in figure 8.

Upon observing the mean accuracy as a function of n for fixed depth quantum layers (figure 8(a)), we observe that, in general (for almost all the depths), the accuracy starts to decline as n increases. This performance decline typically indicates the presence of BP in training landscapes. Based on the results in figure 8(a), we conclude that the BP exists in the local cost function setting also. However, based on the claims in [55], whether the local cost function would lead to an extended depth of quantum layers or not requires analyzing the accuracy trend as a function of depth m (figure 8(b)). Upon observing the mean accuracy as a function of the quantum layer's depth m for a fixed width n, as shown in figure 8(b), we observe that there is no clear improvement in accuracy with an increase in depth m of the underlying quantum layers for a fixed width n. For smaller widths (typically $n = 8$), the allowed depth is slightly greater than for bigger widths ($n = 14$). Moreover, making the cost function local with amplitude encoding resulted in a rather inconsistent performance as compared to the global cost function with amplitude encoding (figure 6).

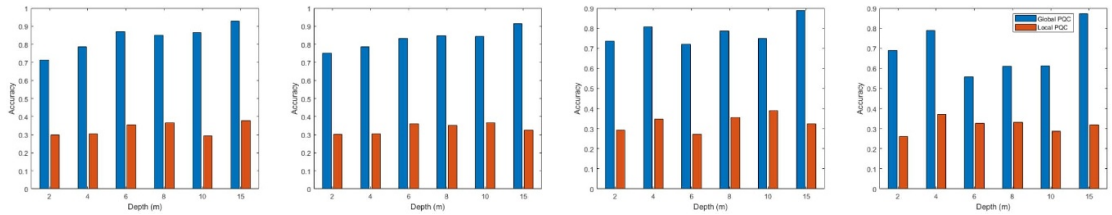
5.1.4. Local PQC with angle encoding

We now analyze the existence of BP for local PQC with angle encoding for multiclass classification. In this case, PCA is applied to reduce the input feature size for all corresponding widths of underlying quantum layers. The mean accuracy for all the experiments performed from figure 5(a) is shown in figure 9. Analyzing the results in figure 9(a), it can be observed that the smallest width quantum layers $n = 8$, for almost all depths, perform slightly better, and in general, a decreasing trend in accuracy with an increase in n is observed indicating towards the presence of BP, similar to that of local PQC with amplitude encoding. Furthermore, for a constant n, the performance deteriorates in general as m increases, as shown in figure 9(b). Again, the accuracy trends do not conform with the solution proposed in [55], which claims to have a delayed occurrence of BP and eventually better learning owing to the greater number of trainable parameters.



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

Figure 9. Multiclass classification—mean accuracy of local PQC with angle encoding.



(a) $n = 8$ (b) $n = 10$ (c) $n = 12$ (d) $n = 14$

Figure 10. Performance comparison of global and local ansatz structures with amplitude encoding.

Based on the results in previous sections, we observe that the global cost function achieves significantly better accuracy than the local cost function. We, therefore, perform a performance comparison of both global and local cost functions for both amplitude and angle encoding.

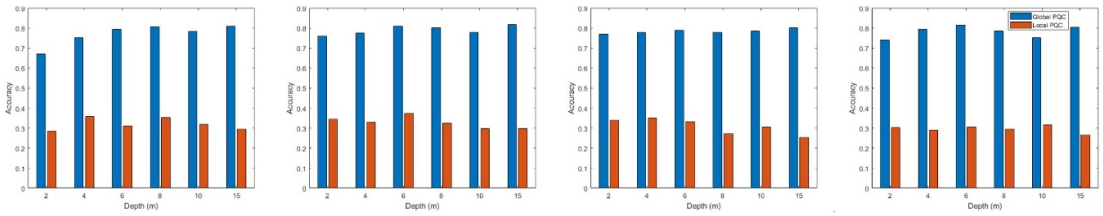
5.2. Multiclass classification: global vs. local PQC

In the previous sections, we discussed the presence of BP for both global and local cost function settings for a multiclass classification task. We also demonstrate that cost functionality does not really help in delaying the BP, which eventually could lead to better performance. The HQNNs, in the case of the local cost function, exhibit a very inconsistent performance irrespective of the encoding routine used. This may be due to the unreliability of cost function locality for multi-class classification problems. However, a brief performance comparison of both global and local ansatz would help better understand the effect of cost function globality/locality. Here, we compare both global and local PQCs to analyze if the local cost function performs any better than the global cost function. We perform the global vs. local PQC's comparison individually for both encodings.

5.2.1. Amplitude encoding

We compare the results of both global and local PQCs when the data is encoded in qubit amplitudes. The performance comparison is conducted in terms of mean accuracy for all the widths and depths of the quantum layers in HQNNs used in this paper and is shown in figure 10.

We observe that the global PQC outperforms local PQC in terms of accuracy in all the experiments, irrespective of the depth and width of underlying quantum layers. Hence the HQNNs with global cost functions are more appropriate than local cost functions. This result contradicts the results of [55], even if the local cost function setting would allow an extended depth of quantum layers due to a significantly lower accuracy achieved (possibly because of BP) as compared to the global cost function setting. Based on the comparison results, we concur that, unlike the claims in [55], which say the global cost function will have BP even for a single layer depth, the global cost function achieves higher accuracy than the local cost function for all the depths and widths.



(a) $n = 8$ (b) $n = 10$ (c) $n = 12$ (d) $n = 14$

Figure 11. Performance comparison of global and local ansatz structures with angle encoding.

5.2.2. Angle encoding

We now compare the performance of both the ansatz structure while encoding the data in qubit rotation angles, typically in *RY* rotations. Analogous to the case of performance comparison for amplitude encoding, the performance is compared in terms of mean accuracy for angle encoding also. The comparison of both global and local PQC for angle encoding is presented in figure 11.

The results in figure 11 clearly show the deteriorated performance in the case of quantum layers with a local cost function, i.e. local PQC. Hence, irrespective of the encoding strategy, the global PQC is better than the local PQC for a multiclass classification problem.

We analyze that in the case of local ansatz, the number of trainable parameters is significantly reduced as compared to all-qubit measurement ansatz because of the lower multiplication factor (n_m) from equation (7). Therefore, to increase the number of parameters, we extended the depth of quantum layers to $m = 15$ (the rightmost bar in figures 10 and 11) for both the encodings, but there is no noticeable improvement in the performance of local PQC even with deeper quantum layers, and in fact, the global PQC, even with minimum depth ($m = 2$) still performs better than the maximum depth of ($m = 15$) in local PQC.

Considering that the BP problem directly results in performance degradation, the comparison results allow us to conclude that a single observable measurement in an n-qubit system is not effective in HQNNs for a multiclass classification application. Furthermore, we speculate that the single-qubit observable measurement approach might be effective in binary classification because of the inherent requirement of a single final observable in binary classification tasks. Therefore, in the following sections, we analyze the impact of cost function globality and locality in binary classification.

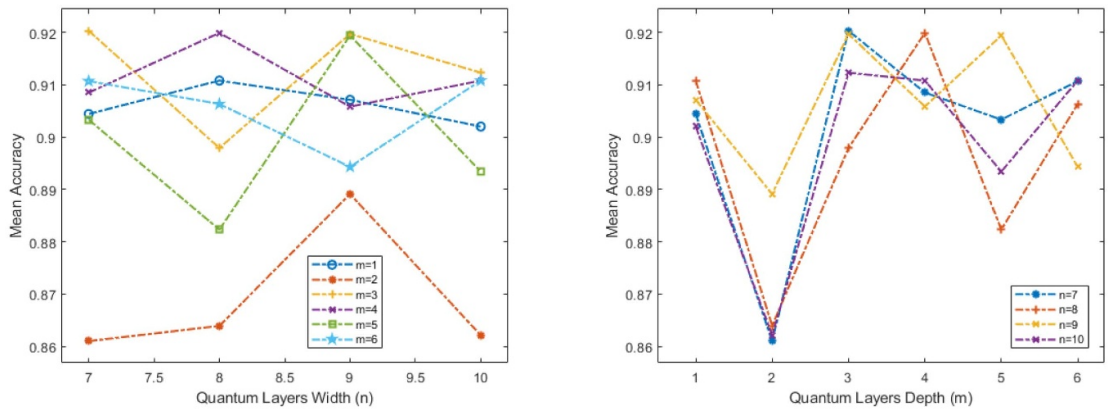
5.3. Binary classification

In this section, we analyze the existence of BP in a binary classification problem. We typically analyze the BP's existence in HQNNs with both global and local cost functions. For the analysis of binary classification, we only consider amplitude encoding for classical to quantum feature mapping because of two reasons; (a) it provides the leverage to have a fixed input size and change the number of qubits, unlike angle encoding, where the input feature size is changed for every change in underlying quantum layers. (b) Given the nature of our analysis which deals with the quantum layer's depth and width, a constant input feature size is more suitable to better analyze the effect of the width and depth of quantum layers.

5.3.1. Global PQC

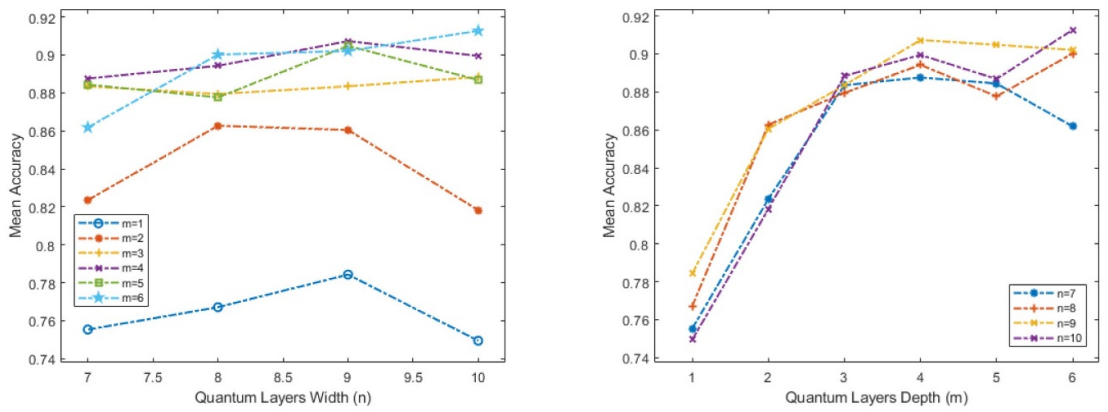
We first analyze the existence of BP in HQNNs when the data is encoded in qubit amplitudes and the cost function is global (global PQC). As discussed in section 4, the input feature dimensionality in the case of binary classification is 30. The minimum number of qubits required to encode these features via amplitude encoding is 5, which is a reasonable number considering NISQ-era limitations. Therefore in binary classification also, we do not apply PCA when encoding the data in qubit amplitudes and the input is directly encoded. A similar list of experiments (as in the case of multiclass classification) is then performed considering different widths (n) and depths (m) of quantum layers, as shown in figure 5(b). The mean accuracy, both as a function of n and m, is shown in figure 12.

By analyzing the mean accuracy as a function of the number of qubits (n), as shown in figure 12(a), we observe that the accuracy declines as n increases. For any fixed depth m in figure 12(a), the performance deteriorates (or has no noticeable improvement) in general as we increase n, which shows the model's exposition to BP. In general, the smaller widths (typically $n = 7$) for almost all depths achieve better accuracy than bigger widths (typically $n = 10$). By analyzing the mean accuracy as a function of the quantum layer's depth (m), as shown in figure 12(b), we observe that for almost all the widths (n), the accuracy starts declining after a certain depth m and in general, smaller n (typically $n = 7, 8$) with moderate depths



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

Figure 12. Binary Classification—mean accuracy of global PQC with amplitude encoding.



(a) Mean accuracy as a function of n (b) Mean accuracy as a function of m

Figure 13. Binary Classification—mean accuracy of local PQC with amplitude encoding.

(typically $m = 3, 4$) yields better performance. However, unlike the case of multiclass classification (figure 6), no *trade-off* between n and m is observed since increasing m for a fixed n, in general, improves the accuracy.

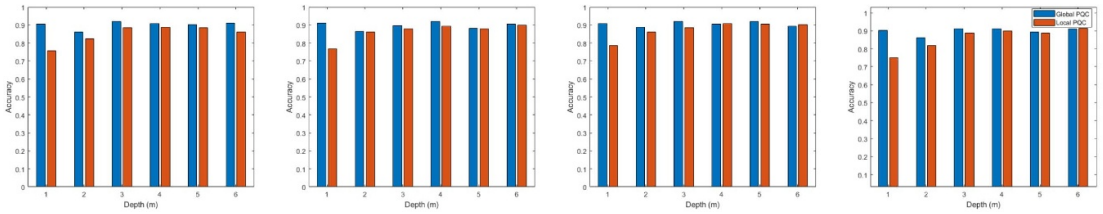
5.3.2. Local PQC

We now analyze the BP's existence in a local cost function setting (local PQC) while encoding the data in qubit amplitudes. The input is directly encoded without PCA. The complete list of experiments, as shown in figure 5(b), is then performed with different n and m. The mean accuracy for all experiments is shown in figure 13.

Analyzing the mean accuracy as a function of the number of qubits (n), we observe that the accuracy for any fixed depth (m) increases (or does not noticeably deteriorate) in general as n increases, which indicates that the model is not (yet) fully exposed to the BP and hence the allowed depth of underlying quantum layers is greater. By analyzing the mean accuracy as a function of the quantum layer's depth (m), we observe that for almost all the widths, the accuracy improves as m increases, unlike the case with the global cost function (figure 12). Hence, the allowed depth is indeed greater in the local cost function setting in conformity with the solution proposed in [55]. As we observed in multiclass classification, the performance in the case of the local cost function is significantly degraded as compared to the global cost function (figures 10 and 11), we now compare both global and local PQC for the binary classification task.

5.4. Binary classification: global vs. local PQC

We now compare the results obtained for both global and local cost function settings. The comparison results are shown in figure 14. Based on the comparison results, we observe that in binary classification tasks, the local cost function setting achieves comparable performance to that of the global cost function setting,



(a) $n = 7$ (b) $n = 8$ (c) $n = 9$ (d) $n = 10$

Figure 14. Binary Classification—performance comparison of global and local PQC with amplitude encoding.

unlike multiclass classification, where it is significantly lower. As shown in figure 14, in almost all cases, the accuracy achieved in local PQC is closer to that of global PQC. However, global PQC is still slightly better than local PQC in terms of accuracy.

Based on the results in figures 12 and 13, we concur that for binary classification tasks, cost function locality does help in achieving an extended depth of quantum layers, as shown in [55]. However, the second claim of [55], which says that the global cost function setting will experience BP even for depth of $\mathcal{O}(1)$, eventually resulting in poor performance than the local cost function setting, contradicts our results where the global cost function still achieves better or equivalent accuracy for all the experiments as shown in figure 14.

6. Conclusion

The phenomenon of BP jeopardizes the practical usability of QNNs. BP is observed when the gradients of parameters vanish exponentially as the system size increases, which adversely impacts the learning process. Recently a solution is proposed in [55] to delay the occurrence of BP up to a certain depth of quantum layers by making it dependent on the cost function. To this end, two cost functions are analyzed. The authors claimed that the global cost function (measuring all qubits in an n-qubit quantum layer) would always result in BP irrespective of the underlying quantum layer's depth. On the other hand, the local cost function (measuring a single qubit in an n-qubit system) can help delay the occurrence of BP up to the depth of $\mathcal{O}(\log(n))$. Since the allowable depth increases in the local cost function setting, as highlighted in [55], it results in a greater number of trainable parameters and is therefore expected to yield better performance.

In this paper, we study the effect of cost function globality and locality in HQNNs for real-world applications, namely, binary and multi-class classification. We consider a single PQC structure both with single (local PQC) and all-qubit (global PQC) measurements. Classical data is encoded in qubit amplitudes and rotation angles. An extensive list of training experiments is then carried out with different widths and depths of hidden quantum layers. We then benchmark accuracy as an evaluation metric for all the experiments. Contrary to [55], our results show that for a multi-class classification problem in a hybrid quantum–classical setting, making the cost function local does not increase the allowable depth of quantum layers. Furthermore, not only the local cost function setting contradicts the solution in [55], but it also results in significantly lower accuracy when compared with the global cost function setting. The lower accuracy in local PQC indicates the possible presence of BP in training landscapes. On the other hand, for the binary classification problem, the local cost function allows having an extended depth of quantum layers when compared to the global cost function, which is in conformity with the results of [55]. However, our results show that the global cost function, in binary classification also, performs slightly better (or equivalent) to that of the local cost function setting for all the depths.

Based on our results, we conclude that using a global cost function setting while dealing with multi-class classification applications yields significantly better results than the local cost function setting. On the other hand, for binary classification problems, both global and local cost functions are equally good. We emphasize that our results are applicable to NISQ devices. A possible future research in this direction could be to analyze the effect of the cost function's globality and locality in HQNNs for other classes of ML problems, such as regression.

Data availability statement

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

Ethical statement

This manuscript does not involve any human or animal participants.

Conflict of interest

The authors declare no competing interests

ORCID iDs

Muhammad Kashif https://orcid.org/0000-0003-2023-6371
Saif Al-Kuwari https://orcid.org/0000-0002-4402-7710

References

[1] Sarker I H 2021 Machine learning: algorithms, real-world applications and research directions *SN Comput. Sci.* **2** 2661–8907
[2] Mohri M, Rostamizadeh A and Talwalkar A 2018 *Foundations of Machine Learning* (Cambridge, MA: MIT press)
[3] Myszczynska M A, Ojamies P N, Lacoste A, Neil D, Saffari A, Mead R, Hautbergue G M, Holbrook J D and Ferraiuolo L 2020 Applications of machine learning to diagnosis and treatment of neurodegenerative diseases *Nat. Rev. Neurol.* **16** 440–56
[4] Sterne J 2017 *Artificial Intelligence for Marketing: Practical Applications* (New York: Wiley)
[5] Awoyemi J O, Adetunmbi A O and Oluwadare S A 2017 Credit card fraud detection using machine learning techniques: a comparative analysis *2017 Int. Conf. on Computing Networking and Informatics (ICCNI)* pp 1–9
[6] Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov R R and Le Q V 2019 Xlnet: generalized autoregressive pretraining for language understanding *Advances in Neural Information Processing Systems* vol 32, ed H Wallach, H Larochelle, A Beygelzimer, F d' Alché-Buc, E Fox and R Garnett (Curran Associates, Inc.)
[7] Kaiming H, Georgia G, Piotr D and Ross G 2017 Mask R-CNN *2017 IEEE Int. Conf. on Computer Vision (ICCV)* pp 2980–8
[8] Xiangnan H, Liao L, Zhang H, Nie L, Xia H and Chua T-S 2017 Neural collaborative filtering *Proc. of the 26th Int. Conf. on World Wide Web, (WWW'17) (Republic and Canton of Geneva, CHE, Int. World Wide Web Conferences Steering Committee)* pp 173–82
[9] Hawkins D M 2004 The problem of overfitting *J. Chem. Inf. Comput. Sci.* **44** 1–12
[10] Daniely A 2016 *Complexity Theoretic Limitations on Learning Halfspaces. (STOC'2016)* (New York: Association for Computing Machinery) pp 105–17
[11] Boob D, Dey S S and Lan G 2022 Complexity of training ReLU neural network *Discret. Optim.* **44** 100620
[12] Arute F *et al* 2019 Quantum supremacy using a programmable superconducting processor *Nature* **574** 505–10
[13] Zhong H-S *et al* 2020 Quantum computational advantage using photons *Science* **370** 1460–3
[14] Wu Y *et al* 2021 Strong quantum computational advantage using a superconducting quantum processor *Phys. Rev. Lett.* **127** 180501
[15] Madsen L S *et al* 2022 Quantum computational advantage with a programmable photonic processor *Nature* **606** 75–81
[16] Schuld M, Sinayskiy I and Petruccione F 2014 An introduction to quantum machine learning *Contemp. Phys.* **56** 172–85
[17] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning *Nature* **549** 195–202
[18] Cerezo M *et al* 2021 Variational quantum algorithms *Nat. Rev. Phys.* **3** 625–44
[19] Huang H-Y, Kueng R and Preskill J 2021 Information-theoretic bounds on quantum advantage in machine learning *Phys. Rev. Lett.* **126** 190505
[20] Huang H-Y, Broughton M, Mohseni M, Babbush R, Boixo S, Neven H and McClean J R 2021 Power of data in quantum machine learning *Nat. Commun.* **12** 2631
[21] Kübler J M, Buchholz S and Schölkopf B 2021 The inductive bias of quantum kernels *Adv. Neural Inf. Process. Syst.* **34** 12661–73 (available at: https://proceedings.neurips.cc/paper/2021/file/69adc1e107f7f7d035d7baf04342e1ca-Paper.pdf)
[22] Abbas A, Sutter D, Zoufal C, Lucchi A, Figalli A and Woerner S 2021 The power of quantum neural networks *Nature Computational Science* **1** 403–9
[23] Date P and Potok T 2021 Adiabatic quantum linear regression *Sci. Rep.* **11** 21905
[24] Arthur D and Date P 2021 Balanced k-means clustering on an adiabatic quantum computer *Quantum Inf. Process.* **20** 294
[25] Rebentrost P, Mohseni M and Lloyd S 2014 Quantum support vector machine for big data classification *Phys. Rev. Lett.* **113** 130503
[26] Havlíček Vĕch, Córcoles A D, Temme K, Harrow A W, Kandala A, Chow J M and Gambetta J M 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209–12
[27] Date P, Arthur D and Pusey-Nazzaro L 2021 QUBO formulations for training machine learning models *Sci. Rep.* **11** 2021
[28] Wan K H, Dahlsten O, Kristjánsson Her, Gardner R and Kim M S 2017 Quantum generalisation of feedforward neural networks *npj Quantum Inf.* **3** 36
[29] Killoran N, Bromley T R, Arrazola J M, Schuld M, Quesada N and Lloyd S 2019 Continuous-variable quantum neural networks *Phys. Rev. Res.* **1** 033063
[30] Beer K, Bondarenko D, Farrelly T, Osborne T J, Salzmann R, Scheiermann D and Wolf R 2020 Training deep quantum neural networks *Nat. Commun.* **11** 1–6
[31] Zoufal C, Lucchi Aelien and Woerner S 2019 Quantum generative adversarial networks for learning and loading random distributions *npj Quantum Inf.* **5** 103
[32] Kamruzzaman A, Alhwaiti Y, Leider A and Tappert C C 2020 *Future of information and communication conf.* (*Cham*) pp 299–311
[33] Arthur D and Date P 2022 A hybrid quantum-classical neural network architecture for binary classification (arXiv:2201.01820)
[34] Bergholm V *et al* 2018 Pennylane: automatic differentiation of hybrid quantum-classical computations (arXiv:1811.04968)
[35] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001
[36] Broughton M *et al* 2020 Tensorflow quantum: a software framework for quantum machine learning
[37] Sim S, Johnson P D and Aspuru-Guzik Aan 2019 Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms *Adv. Quantum Technol.* **2** 1900070

[38] Hubregtsen T, Pichlmeier J, Stecher P and Bertels K 2021 Evaluation of parameterized quantum circuits: on the relation between classification accuracy, expressibility and entangling capability *Quantum Mach. Intell.* **3** 1–19
[39] Schuld M and Petruccione F 2018 *Supervised Learning With Quantum Computers* vol 17 (Berlin: Springer)
[40] Andrea Mari T R Bromley J I, Schuld M and Killoran N 2020 Transfer learning in hybrid classical-quantum neural networks *Quantum* **4** 340
[41] Verdon G, Broughton M and Biamonte J 2017 A quantum algorithm to train neural networks using low-depth circuits (arXiv:1712.05304)
[42] Benedetti M, Garcia-Pintos D, Perdomo O, Leyton-Ortega V, Nam Y and Perdomo-Ortiz A 2019 A generative modeling approach for benchmarking and training shallow quantum circuits *npj Quantum Inf.* **5** 45
[43] Liu J-G and Wang L 2018 Differentiable learning of quantum circuit born machines *Phys. Rev.* A **98** 062324
[44] Coyle B, Mills D, Danos V and Kashefi E 2020 The born supremacy: quantum advantage and training of an ising born machine *npj Quantum Inf.* **6** 60
[45] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324
[46] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)
[47] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 Circuit-centric quantum classifiers *Phys. Rev.* A **101** 032308
[48] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 Hierarchical quantum classifiers *npj Quantum Inf.* **4** 1–8
[49] Pérez-Salinas Aan, Cervera-Lierta A, Gil-Fuster E and Latorre Je I 2020 Data re-uploading for a universal quantum classifier *Quantum* **4** 226
[50] Blank C, Park D K, Rhee J-K K and Petruccione F 2020 Quantum classifier with tailored quantum kernel *npj Quantum Inf.* **6** 1–7
[51] Kashif M and Al-Kuwari S 2021 Design space exploration of hybrid quantum–classical neural networks *Electronics* **10** 2980
[52] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273–8
[53] Arrasmith A, Holmes Z, Cerezo M and Coles P J 2022 *Quantum Sci. Technol.* **7** 045015
[54] McClean J R, Boixo S, Smelyanskiy V N, Babbush R and Neven H 2018 Barren plateaus in quantum neural network training landscapes *Nat. Commun.* **9** 4812
[55] Cerezo M, Sone A, Volkoff T, Cincio L and Coles P J 2021 Cost function dependent barren plateaus in shallow parametrized quantum circuits *Nat. Commun.* **12** 1791
[56] Wierichs D, Gogolin C and Kastoryano M 2020 Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer *Phys. Rev. Res.* **2** 043246
[57] Wang S, Enrico Fontana M Cerezo K S, Sone A, Cincio L and Coles P J 2021 Noise-induced barren plateaus in variational quantum algorithms *Nat. Commun.* **12** 6961
[58] Enrico Fontana M Cerezo A A, Rungger I and Coles P J 2020 Optimizing parametrized quantum circuits via noise-induced breaking of symmetries
[59] Wang S, Czarnik P, Arrasmith A, Cerezo M, Cincio L and Coles P J 2021 Can error mitigation improve trainability of noisy variational quantum algorithms? (arXiv:2109.01051)
[60] França D S and Garcia-Patron R 2021 Limitations of optimization algorithms on noisy quantum devices *Nat. Phys.* **17** 1221–7
[61] Sharma K, Cerezo M, Cincio L and Coles P J 2022 Trainability of dissipative perceptron-based quantum neural networks *Phys. Rev. Lett.* **128** 180505
[62] Renes J M, Blume-Kohout R, Scott A J and Caves C M 2004 Symmetric informationally complete quantum measurements *J. Math. Phys.* **45** 2171–80
[63] Harrow A W and Low R A 2009 Random quantum circuits are approximate 2-designs *Commun. Math. Phys.* **291** 257–302
[64] Maciejewski F B, Baccari F, Zimborás Zan and Oszmaniec Mł 2021 Modeling and mitigation of cross-talk effects in readout noise with applications to the quantum approximate optimization algorithm *Quantum* **5** 464
[65] Preskill J 2018 Quantum Computing in the NISQ era and beyond *Quantum* **2** 79
[66] Alam M, Ash-Saki A and Ghosh S 2019 Analysis of quantum approximate optimization algorithm under realistic noise in superconducting qubits (arXiv:1907.09631)
[67] Xue C, Chen Z-Y, Yu-Chun W and Guo G-P 2021 Effects of quantum noise on quantum approximate optimization algorithm *Chin. Phys. Lett.* **38** 030302
[68] Marrero C O, Kieferová M and Wiebe N 2021 Entanglement induced barren plateaus *PRX Quantum* **2** 040316
[69] Volkoff T and Coles P J 2021 Large gradients via correlation in random parameterized quantum circuits *Quantum Sci. Technol.* **6** 025008
[70] Skolik A, McClean J R, Mohseni M, van der Smagt P and Leib M 2021 Layerwise learning for quantum neural networks *Quantum Mach. Intell.* **3** 5
[71] Holmes Z, Sharma K, Cerezo M and Coles P J 2022 Connecting ansatz expressibility to gradient magnitudes and barren plateaus *PRX Quantum* **3** 010313
[72] Schuld M, Sweke R and Meyer J J 2021 Effect of data encoding on the expressive power of variational quantum-machine-learning models *Phys. Rev.* A **103** 032430
[73] Lloyd S, Schuld M, Ijaz A, Izaac J and Killoran N 2020 Quantum embeddings for machine learning (arXiv:2001.03622)
[74] Romero J, Olson J P and Aspuru-Guzik A 2017 Quantum autoencoders for efficient compression of quantum data *Quantum Sci. Technol.* **2** 045001
[75] LaRose R and Coyle B 2020 Robust data encodings for quantum classifiers *Phys. Rev.* A **102** 032420
[76] Cao S, Wossnig L, Vlastakis B, Leek P and Grant E 2020 Cost-function embedding and dataset encoding for machine learning with parametrized quantum circuits *Phys. Rev.* A **101** 052309
[77] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309
[78] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 Evaluating analytic gradients on quantum hardware *Phys. Rev.* A **99** 032331
[79] Pedregosa F *et al* 2011 Scikit-learn: Machine Learning in Python **12** 2825–30