

Coreset selection can accelerate quantum machine learning models with provable generalization

Yiming Huang^{*} and Xiao Yuan[†]


*Center on Frontiers of Computing Studies, Peking University, 100871 Beijing, China
School of Computer Science, Peking University, 100871 Beijing, China*

Huiyuan Wang[‡]

Peterhouse, Univeristy of Cambridge, Cambridge, Cambridgeshire, CB2 1RD, United Kingdom

Yuxuan Du[§]

JD Explore Academy, 101111 Beijing, China

 (Received 11 December 2023; revised 16 April 2024; accepted 27 June 2024; published 29 July 2024)

Quantum neural networks (QNNs) and quantum kernels stand as prominent figures in the realm of quantum machine learning, poised to leverage the nascent capabilities of near-term quantum computers to surmount classical machine learning challenges. Nonetheless, the training-efficiency challenge poses a limitation on both QNNs and quantum kernels, curbing their efficacy when they are applied to extensive datasets. To confront this concern, we present a unified approach—coreset selection—aimed at expediting the training of QNNs and quantum kernels by distilling a judicious subset from the original training dataset. Furthermore, we analyze the generalization-error bounds of QNNs and quantum kernels when they are trained on such coresets, unveiling performance comparable with that of those trained on the complete original dataset. Through systematic numerical simulations, we illuminate the potential of coreset selection in expediting tasks encompassing synthetic data classification, identification of quantum correlations, and quantum compiling. Our work offers a useful way to improve diverse quantum machine learning models with a theoretical guarantee while reducing the training cost.

DOI: [10.1103/PhysRevApplied.22.014074](https://doi.org/10.1103/PhysRevApplied.22.014074)

I. INTRODUCTION

Quantum neural networks (QNNs) [1–4] and quantum kernels [5,6] have emerged as pivotal models in the burgeoning field of quantum machine learning (QML) [7–9], poised to unlock the power of near-term quantum computers to address challenges that elude classical machine learning paradigms [10,11]. The allure of these models is rooted in a fusion of theoretical advances and practical adaptability. That is, theoretical evidence showcases their superiority over classical counterparts in diverse scenarios, spanning synthetic datasets, discrete logarithmic problems, and quantum information processing tasks [6,12–16], as measured by sample complexity and runtime considerations. Complementing their theoretical strength, their implementation displays flexibility, adeptly accommodating constraints posed by contemporary quantum hardware, including qubit connectivity and limited circuit depth.

This convergence of theoretical promise and practical flexibility has spurred a wave of empirical investigations, substantiating the viability and potential benefits of QNNs and quantum kernels across real-world applications such as computer vision [17–19] and quantum physics [20–28].

Despite their promising potential, QNNs and quantum kernels face a pertinent challenge concerning the training efficiency, resulting in a constrained practical applicability towards large-scale datasets [29]. This limitation is particularly evident due to the absence of fundamental training mechanisms such as back-propagation and batch gradient descent in most QNNs, which are imperative for the swift training of deep neural networks [30]. Similarly, the training process for quantum kernels necessitates the collection of a kernel matrix of size $O(N^2)$, with N being the number of training examples and each entry demanding independent evaluation via a specific quantum circuit. Consequently, the capacities of both QNNs and quantum kernels to effectively navigate vast training datasets, characterized by millions of data points, are compromised.

In response to the above-mentioned challenge, several research lines have emerged to increase the training

^{*}Contact author: yiminghwang@gmail.com

[†]Contact author: xiaoyuan@pku.edu.cn

[‡]Contact author: hw531@cam.ac.uk

[§]Contact author: duyuxuan123@gmail.com

efficiency of QNNs. The first line embarks on improving the optimizer or the initialization methods, seeking to expedite convergence towards the minimal empirical risk through a reduction in measurements and iterations [31–35]. Nonetheless, the nonconvex nature of the loss landscape cautions against potential entrapment within saddle points during the optimization process [36–38]. The second avenue delves into feature-dimension-reduction techniques, enabling more-streamlined use of quantum resources for each data point [39,40]. However, this approach may not necessarily alleviate overall runtime complexity, potentially even exacerbating it. The third research pathway navigates the realm of QNN expressivity, imposing judicious constraints to facilitate the integration of back-propagation through meticulously engineered ansatzes [41,42]. Nevertheless, the extent to which these constrained ansatzes might impact QNN performance on unseen data remains a dynamic yet-unresolved facet.

The endeavor to increase the training efficiency of quantum kernels has received less attention, in contrast to efforts to increase the training efficiency of QNNs. This divergence in attention stems from the shared observation that both classical kernels and quantum kernels demand $O(N^2)$ runtime for the collection of the kernel matrix. Existing literature focused on increasing the training efficiency of quantum kernels has predominantly centered on the development of advanced encoding methods [43,44]. These methods aim to mitigate the use of quantum resources and attenuate the manifestation of barren plateaus [45]. However, despite the cultivation of various research trajectories directed at enhancing QML models, it is noteworthy that these approaches often retain model-specific attributes, potentially harboring unforeseen side effects. This realization begets a pivotal inquiry: can a unified approach be devised that systematically increases the training efficiency of both QNNs and quantum kernels while safeguarding a steadfast theoretical guarantee?

In this study, we provide an affirmation of the above question by introducing coreset-selection techniques into QML. Conceptually, coreset selection is an effective preprocessing approach to distill a weighted subset from the large training dataset, which guarantees that models fitting the coreset also provide a good fit for the original data. Considering that the training efficiency of both QNNs and quantum kernels hinges on the number of training examples, coreset selection provides a unified way to increase their training efficiency. Moreover, the theoretical foundation of coreset selection is established on the recent generalization-error analysis of QNNs and quantum kernels, indicating that a few training examples are sufficient to achieve a good test performance. In this regard, we provide a rigorous analysis of the generalization ability of QNNs and quantum kernels trained on the coreset. The bounds achieved exhibit comparable generalization

performance of QNN and quantum kernel learning when they are optimized under the original dataset and the coreset. Numerical simulations on synthetic data, identification of nonclassical correlations in quantum states, and quantum circuit compilation confirm the effectiveness of our proposal.

II. RELATED WORK

The prior literature related to our work can be divided into two classes: algorithms for accelerating the optimization of QML models, and the generalization-error analysis of QML models. In the following, we separately explain how our work relates to and differs from previous studies.

A. Acceleration algorithms for QML models

As already mentioned, various algorithms have been introduced to expedite the optimization of QNNs rather than quantum kernels. These algorithms can be classified into three distinct categories, each addressing different facets of optimization improvement: data feature engineering, QNN-architecture design, and optimizer enhancement.

In the realm of data feature engineering, the core concept revolves around implementing dimensionality-reduction techniques such as principal-component analysis and feature selection during the preprocessing stage [46]. This strategy effectively reduces the quantum resources required for processing data points compared with their unprocessed counterparts. Within the domain of architecture design, there exists a dual focus on encoding-strategy design [47,48] and ansatz design [49–53]. The underlying principle emphasizes the use of a minimal number of qubits, trainable parameters, and shallower circuit depths, all geared towards achieving competent performance in learning tasks. In parallel, efforts to enhance optimizers have also attracted attention. The deployment of higher-order gradient-descent or machine learning–assisted optimizers [31–35] and distributed-learning schemes [54] has been advocated as a means to accelerate convergence rates and wall-clock time, thereby further increasing the efficiency of QNN optimization.

Our work is complementary to the above-mentioned approaches since the reduced size of the data is independent of data feature engineering, QNN-architecture design, and optimizer enhancement. In other words, QNNs trained on the coreset can be further accelerated by the above-mentioned approaches. Moreover, differently from prior literature that concentrates on the acceleration of QNNs, coreset selection can be directly used to accelerate the collection of quantum kernels.

B. Generalization of QML models

Several studies have undertaken the task of quantifying the generalization error of QNNs through the lens of the foundational learning-theoretic technique known as uniform convergence [55–59]. In general, the resulting bound for generalization adheres to the format of $O(\sqrt{p/N})$, where p denotes the number of trainable parameters and N signifies the number of training instances. For quantum kernels, the generalization-error upper bound scales with $O(\sqrt{C_1/N})$, where C_1 depends on the labels of the data and the value of the quantum kernel [6]. When system noise is considered, the generalization-error upper bound degrades to $O(\sqrt{C_1/N} + N/(C_2\sqrt{m}))$, where m is the shot number and C_2 depends on the kernel value, shot number, and noise level [60].

Our work leverages the above analysis to quantify how the coreset selection affects the learning performance of QNNs and quantum kernels, respectively.

We note that although Ref. [61] also discusses quantum coreset, its primary emphasis lies in using fault-tolerant quantum computers to speed up the construction of coresets, a subject that falls outside the scope of our work. In addition, Refs. [62,63] illustrated the potential of coreset in specific applications, i.e., clustering and image classification, without a unified view and generalization-error analysis.

III. PRELIMINARIES

In this section, we first introduce the concept of machine learning and a coreset. Then we formally introduce the mechanism of QNNs and quantum kernels under the supervised-learning paradigm.

A. Foundations of machine learning

Let $\mathcal{S}_t = \{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$ be the dataset in which each paired training example (\mathbf{x}_i, y_i) is independent and identically sampled over $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ with probability distribution $p_{\mathcal{Z}}$, where \mathcal{X} is the feature space and \mathcal{Y} is the label space. The goal of supervised-learning algorithms is to find a hypothesis $f : \mathcal{X} \rightarrow \mathcal{Y}$ with trainable parameters \mathbf{w} such that the true risk R on the distribution $p_{\mathcal{Z}}$ is minimized with

$$R = \mathbb{E}_{(\mathbf{x}, y) \sim p_{\mathcal{Z}}} [l(f_{\mathbf{w}}(\mathbf{x}), y)], \quad (1)$$

where $l(\cdot, \cdot)$ is the loss function used to measure the degree of fit between the output of the hypothesis and its corresponding ground truth. As the distribution $p_{\mathcal{Z}}$ is unknown and given that accessing all the data over \mathcal{Z} would be impractical, in practice, the optimal hypothesis is estimated by one optimizing \mathbf{w} to minimize the empirical risk R_e over the training dataset, i.e.,

$$R_e = \frac{1}{N_t} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{S}_t} l(f_{\mathbf{w}}(\mathbf{x}_i), y_i). \quad (2)$$

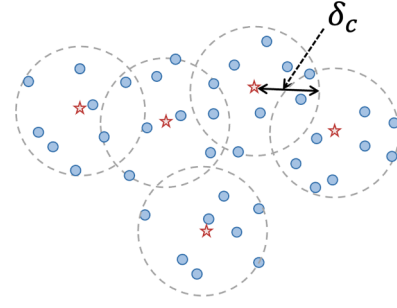


FIG. 1. Coreset construction as a k -center problem. The red stars are k picked points with radius δ_c covering the entire set. In the case shown, there are five center data points $P_k \in \mathcal{S}$, with $k \in \{1, 2, \dots, 5\}$, such that the maximum distance from any point in \mathcal{D} to its closest center is minimized.

B. Coreset in machine learning

As the learning algorithm evolves, not only are models becoming increasingly complex, but training datasets are also growing larger. With growing volumes of data, the challenges for how to organize and analyze the massive data force us to devise effective approaches to condense the enormous dataset. The concept of coreset selection, as a paradigm for extracting a data subset that comprises informative samples such that the generalization performance of the model trained on this reduced set is close to that of models trained on the entire dataset [64], is a promising solution to address the issue.

Definition (Coreset). Let P be a set of points in space \mathcal{V} , and let f be a monotone measure function. We call a subset $Q \subseteq P$ an ϵ coreset of P if

$$|f(Q) - f(P)| \leq \epsilon \cdot f(P). \quad (3)$$

Various coreset-selection approaches are used to assist in dealing with computationally intractable problems in different learning tasks and data types [65,66]. Throughout the whole work, we consider a geometry-based method to construct the coreset [67]. As shown in Fig. 1, the goal of coreset construction is to find k data points as the centers \mathcal{C} such that the maximum distance between any point $s \in \mathcal{S}$ and its nearest center is minimized, i.e., selecting \mathcal{C} such that the radius δ_c minimized. The optimization of finding the coreset can be formulated as

$$\mathcal{S}_c = \arg \min_{\mathcal{C} \subseteq \mathcal{S}, |\mathcal{C}|=k} \max_{\mathbf{x}_j \in \mathcal{S}} D(\mathbf{x}_j, \mathbf{x}_c), \quad (4)$$

where $D(\mathbf{x}_i, \mathbf{x}_c) = \min_{\mathbf{x}_c \in \mathcal{C}} d(\mathbf{x}_i, \mathbf{x}_c)$ denotes the distance between point i and its closest center. Although it is a non-deterministic polynomial time-hard problem to find \mathcal{S}_c , there is a provable greedy algorithm that can efficiently get a 2-approximate solution, i.e., if \mathcal{C}^* is the optimal solution of Eq. (4), we can efficiently find a solution \mathcal{C} such that $\delta_{\mathcal{C}^*} \leq \delta_{\mathcal{C}} \leq 2 \cdot \delta_{\mathcal{C}^*}$.

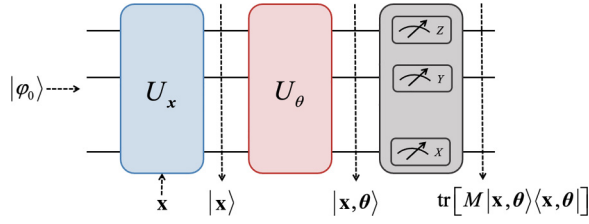


FIG. 2. A general feed-forward process of quantum neural networks. The input state $|\phi_0\rangle$ is firstly fed to a feature-encoding block to map the classical data \mathbf{x} to $|\mathbf{x}\rangle$, and then the variational circuit U_θ is used to form a parametrized state $|\mathbf{x}, \theta\rangle$, which is used to minimize the loss function. In the end, the measurement output is used to estimate the prediction residuals.

C. Quantum neural networks

“Quantum neural network” refers to a class of neural networks that leverages the power of variational quantum circuits and classical optimizers to tackle learning problems. A QNN is composed mainly of three components—feature encoding, a variational parametrized circuit, and measurement—as depicted in Fig. 2. Generally, feature encoding uses an encoding circuit U_x to map the classical input \mathbf{x} into an n -qubit state $|\mathbf{x}\rangle$. The concrete approaches of feature encoding are diverse, as outlined in Refs. [48,68]. A variational parametrized circuit U_θ evolves the feature state $|\mathbf{x}\rangle$ to a parametrized state $|\mathbf{x}, \theta\rangle$, where the parameters θ are to be tuned to minimize the training loss. Measurement extracts the processed information stored in the parametrized state $|\mathbf{x}, \theta\rangle$ into the classical register and may be combined with a postprocessing operation to form the output of the QNN.

In this work, we consider a QNN implemented by the data-reuploading strategy [47,68,69], alternating the feature-encoding circuit U_x and the variational circuit U_θ to generate the parametrized state $|\mathbf{x}, \theta\rangle$, i.e.,

$$|\mathbf{x}, \theta\rangle = U_\theta U_x \cdots U_x U_\theta U_x |\phi_0\rangle. \quad (5)$$

Without loss of generality, the feature-encoding circuit and the variational circuit take the forms

$$U_x = \bigotimes_{j=1}^n \exp(-ix_j H) \text{ and } U_\theta = \bigotimes_{j=1}^n \exp(-i\theta_j H) V, \quad (6)$$

where $H \in \{\sigma_x, \sigma_y, \sigma_z\}$ is the Hermitian operator and V refers to the fixed gates such as a sequence of controlled-NOT gates. Once the variational state $|\mathbf{x}, \theta\rangle$ is evolved, the measurement operator M is applied to obtain the estimated expectation value. Given the training dataset $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$, the empirical risk of the QNN over \mathcal{S} is given by

$$R_e^{\text{QNN}}(\theta) = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x}_i, y_i \in \mathcal{S}} l(\text{tr}[M|\mathbf{x}_i, \theta\rangle\langle\mathbf{x}_i, \theta|], y_i). \quad (7)$$

A possible choice of l is the mean square error, i.e., $l(a, b) = (a - b)^2$. The optimization of the QNN, i.e., the minimization of R_e^{QNN} , can be completed by the gradient-descent optimizer based on the parameter-shift rule [29].

D. Quantum kernels

Kernel methods have been extensively studied in classical machine learning and have been applied to various scenarios, such as face recognition and the interpretation of the dynamics of deep neural networks [70,71]. Their quantum counterparts have also been extensively investigated from both experimental and theoretical perspectives [3,60]. Formally, quantum kernels leverage quantum feature maps that encode the classical vector \mathbf{x} into a higher Hilbert space to perform the kernel trick. One well-known quantum kernel function $\kappa(\mathbf{x}, \mathbf{x}')$ is defined as the overlap between the quantum states $|\mathbf{x}\rangle$ and $|\mathbf{x}'\rangle$ that encodes \mathbf{x} and \mathbf{x}' via an n -qubit feature-encoding circuit U_e , i.e., $\kappa(\mathbf{x}, \mathbf{x}') = |\langle\mathbf{x}'|\mathbf{x}\rangle|^2$, with $|\mathbf{x}\rangle = U_e(\mathbf{x})|+\rangle^{\otimes N}$ and $|+\rangle = H|0\rangle$, as shown in Fig. 3. In this work, we consider a generic quantum feature map as proposed in Ref. [3], i.e.,

$$U_e(\mathbf{x}) = \exp\left(\sum_i x_i \sigma_j^Z + \sum_{ij} (\pi - x_i)(\pi - x_j) \sigma_i^Z \sigma_j^Z\right). \quad (8)$$

We note that other symmetric positive-definite functions are also valid candidates for implementing quantum kernels. Different forms of the quantum kernel correspond to different feature maps in Hilbert space, which could provide quantum merits for classification tasks if designed properly [14].

IV. CORESET SELECTION FOR QML MODELS

In this section we first introduce the algorithmic implementation of coreset selection. Then we elucidate how to use the constructed coreset to train QNNs and quantum kernels.

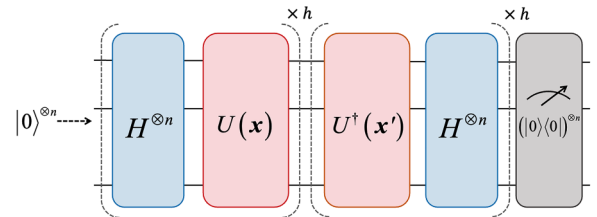


FIG. 3. Quantum circuit implementation of the quantum kernel $\kappa(\mathbf{x}, \mathbf{x}')$ considered in this work. The circuit separately encodes \mathbf{x} and \mathbf{x}' as the parameters of h -layer variational circuits U and U^\dagger into the states $|\mathbf{x}\rangle$ and $|\mathbf{x}'\rangle$.

ALGORITHM 1. The greedy algorithm for k -center coreset selection.

```

input : Data set  $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{N_t}, y_{N_t})\}$ ,
        covering number  $k$ .
output: The subset  $\mathcal{S}_c \subseteq \mathcal{S}$  with  $|\mathcal{S}_c| = k$  and
        corresponding weight  $\{\gamma_s\}_{s=1}^k$ .

1 for  $i \leftarrow 1$  to  $n_c$  do
2    $\mathcal{S}_c^{(i)} \leftarrow \emptyset$ ; /* initialize the coreset for
   each class */
3 end for
4 for  $i \leftarrow 1$  to  $n_c$  do
5    $\mathcal{S}_c^{(i)} \leftarrow \mathbf{x} \in \mathcal{S}^{(i)}$ ;
6   /* randomly select point  $\mathbf{x}$  from set  $\mathcal{S}^{(i)}$ 
   associated with class  $i$ , and add it
   to set  $\mathcal{S}_c^{(i)}$ . */
7   while  $|\mathcal{S}_c^{(i)}| < \lceil \frac{|\mathcal{S}^{(i)}|}{|\mathcal{S}|} \cdot k \rceil$  do
8      $\mathbf{x} = \arg \max_{\mathbf{x} \in \mathcal{S}^{(i)} \setminus \mathcal{S}_c^{(i)}} \min_{\mathbf{x}'} d(\mathbf{x}, \mathbf{x}')$ ;
9      $\mathcal{S}_c^{(i)} \leftarrow \mathcal{S}_c^{(i)} \cup \{\mathbf{x}\}$ ;
10  end while
11 end for
12  $s = 1$ ;
13 for  $j \leftarrow 1$  to  $n_c$  do
14   for  $\mathbf{x} \in \mathcal{S}_c^{(j)}$  do
15      $\gamma_s \leftarrow \sum_{\mathbf{x}' \in \mathcal{S}^{(j)}} I_{|\mathbf{x} - \mathbf{x}'| \leq \delta_c^{(j)}}$ ;
16     /* use the indicator function to
     count the number of same-class
     training samples in radius  $\delta_c$ .
     */
17      $s \leftarrow s + 1$ ;
18   end for
19 end for
20  $\mathcal{S}_c \leftarrow \emptyset$ ;
21 for  $i \leftarrow 1$  to  $n_c$  do
22    $\mathcal{S}_c \leftarrow \mathcal{S}_c \cup \mathcal{S}_c^{(i)}$ 
23 end for
24 return  $\mathcal{S}_c$  and  $\{\gamma_s\}_{s=1}^k$ .

```

The pseudocode of coreset selection for QML models is summarized in Algorithm 1. The main idea is regarding the coreset selection as a data-preprocessing strategy, which aims to find k balls, centered at points $\{\mathbf{x}_s\}_{s=1}^k$ in data space \mathcal{P} , to cover the whole dataset with radius δ_c . Let $\mathcal{S} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{N_t}, y_{N_t})\}$ be an n_c -class dataset and let a hyperparameter k satisfy $k \gg n_c$. We apply the following procedure to each class to construct the coreset.

For each class $i \in [n_c]$, in the first step, we randomly pick a data point \mathbf{x} from the set $\mathcal{S}^{(i)} \subseteq \mathcal{S}$ and put it into the initialized empty set $\mathcal{S}_c^{(i)}$, where $\mathcal{S}^{(i)}$ refers to the set of all training data points associated with the label i . In the second step, we iteratively choose the data point from $\mathcal{S}^{(i)}$ to be as far away as possible from the other centers in $\mathcal{S}_c^{(i)}$. That is, for each class i , we repeatedly find a data point \mathbf{x} for which the distance $d(\mathbf{x}, \mathbf{x}')$ is maximized, where $\mathbf{x} \in \mathcal{S}^{(i)}$, $\mathbf{x}' \in \mathcal{S}_c^{(i)}$. The searched-for data point is

then appended to $\mathcal{S}_c^{(i)}$. This iteration procedure is terminated when $|\mathcal{S}_c^{(i)}| \geq \lceil (|\mathcal{S}^{(i)}|/|\mathcal{S}|) \cdot k \rceil$. In other words, the coreset size of each class i is restricted to be proportional to the ratio of the size of that class $|\mathcal{S}^{(i)}|$ to the overall amount of data $|\mathcal{S}|$. When the coreset for each class is collected, we merge these sets to create a coreset with $\mathcal{S}_c = \bigcup_{i=1}^{n_c} \mathcal{S}_c^{(i)}$ and set the weight γ_s as the number of samples covered by the coreset example \mathbf{x}_s in radius δ_c . Once the coreset \mathcal{S}_c is built, we can integrate it into QML models.

We note that k is a hyperparameter that depends on multiple factors, such as the underlying data distribution and the trade-off between accuracy and coreset size [72,73]. A crucial research line of coreset construction is to determine an appropriate k , and any advanced method for this topic can be seamlessly embedded into our framework. Typical instances include geometric decomposing-based coreset algorithms [72,74–76]. In addition, it was pointed out in Ref. [76] that if the loss function of the learning problem satisfies certain continuous conditions, there are some fundamental properties of the coreset generated by k clustering. As a result, we can heuristically find k by following Algorithm 1 given in Ref. [76].

Remark. As we use the fidelity-based distance to measure the similarity between data points which scales with $\mathcal{O}(n^2)$, it becomes a bottleneck for pruning the dataset. The training efficiency also brings a limitation to selecting a coreset; that is, the proposed method is based on a greedy algorithm that takes a relatively long time to go through the entire dataset to find out k . Nevertheless, such a pruning process needs to be performed only once, and the preprocessing requires just a single execution, after which the resultant coreset can be used indefinitely for various tasks, such as model training, parameter tuning, and quantum ansatz design, which leads to decreased computational and storage requirements. In addition, we believe more-efficient similarity-estimation techniques will be proposed in the future to address the cost issue of similarity estimation, whether in classical or quantum.

ALGORITHM 2. QNN with coreset selection.

```

input : Coreset  $\mathcal{S}_c$ , weights  $\{\gamma_s\}_{s=1}^k$ , maximum
        epoch number  $K_{max}$ , learning rate  $\eta$ 
output: trained QNN.

1 Initialize the parametrized quantum circuit  $U(\theta)$ ;
2  $l = 1$ ;
3 while  $l < K_{max}$  do
4   for  $\mathbf{x}_j, y_j \in \mathcal{S}_c$  do
5     Estimate the gradient of  $R_c$  with respect
     to  $\theta$ ,  $\nabla_{\theta} f(\mathbf{x}_j, y_j)$ ;
6      $\theta_{l+1} \leftarrow \theta_l + \eta \cdot \gamma_j \cdot \nabla_{\theta} f(\mathbf{x}_j, y_j)$ ;
7   end for
8 end while
9 return the QNN  $tr[M \cdot U(\theta)U(\mathbf{x}_i)\rho_0 U^{\dagger}(\theta)U^{\dagger}(\mathbf{x}_i)]$ .

```

 ALGORITHM 3. Quantum kernel with coresets selection.

input : Coreset \mathcal{S}_c , weights $\{\gamma_s\}_{s=1}^k$, encoding circuit U , regularization parameter C
output: SVM with quantum kernel.

```

1 for  $\mathbf{x} \in \mathcal{S}_c$  do
2   for  $\mathbf{x}' \in \mathcal{S}_c$  do
3      $K_{\mathbf{x}, \mathbf{x}'} = |\langle 0|U^\dagger(\mathbf{x}')|U(\mathbf{x})|0\rangle|^2$ ;
4     /* construct kernel matrix  $K$  */
5   end for
6 end for
7 Solve the dual problem in Eq.(10), get the
  optimal parameters  $\alpha^*$ ;
8 Calculate the parameter  $W^*, b^*$ ;
```

$$W^* = \sum_i \alpha_i y_i \mathbf{x}_i, \quad (11)$$

$$b^* = y_i - \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j), \quad (12)$$

return the quantum kernel based SVM,
 $f(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b^*$.

In the following, we introduce how to apply the proposed coreset to QNNs and quantum kernels, respectively.

The pseudocode of a QNN trained on the coreset is summarized in Algorithm 2. Conceptually, we only need to replace the full training data \mathcal{S} with the coreset \mathcal{S}_c , and rewrite the cost function by introducing the weight $\{\gamma_s\}$ for each corresponding data point \mathbf{x} in the coreset \mathcal{S}_c , i.e.,

$$R_c^{\text{QNN}} = \frac{1}{|\mathcal{S}_c|} \sum_{\mathbf{x}_s, y_s \in \mathcal{S}_c} \gamma_s \cdot l(\text{tr}[M \cdot |\mathbf{x}_s, \theta\rangle\langle \mathbf{x}_s, \theta|, y_s]), \quad (9)$$

where $|\mathbf{x}_s, \theta\rangle$ and M are as in Eq. (7).

We next explain the implementation of a quantum kernel-based support-vector-machine (SVM) classifier on the coreset. As we use the coreset \mathcal{S}_c and introduce the weights $\{\gamma\}$, there is a slight difference between the original SVM and the coreset-enhanced SVM, where the Lagrange multipliers α_i in the dual problem are upper bounded by $C \cdot \gamma_i$ instead of C . Mathematically, we have

$$\begin{aligned} \max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j), \\ \text{subject to} \quad & \sum_i y_i \alpha_i = 0, \\ & 0 \leq \alpha_i \leq C \cdot \gamma_i, \quad i = 1, \dots, k, \end{aligned} \quad (10)$$

where $K(\mathbf{x}, \mathbf{x}') = |\langle \mathbf{x} | \mathbf{x}' \rangle|^2$ is the quantum kernel function. The training process is similar to that for the original SVM and is given in Algorithm 3.

V. GENERALIZATION ABILITY OF QML MODELS UNDER CORESET SELECTION

In machine learning, generalization analysis is a crucial theoretical tool to measure how accurately a learning model predicts previously unseen data [77]. It helps us as a guiding metric to choose the best-performing model when comparing different model variations. As explained in Sec. III A, the purpose of learning algorithms is to find a hypothesis f such that the true risk R on the data distribution \mathcal{Z} is as small as possible. However, it is unlikely to directly estimate the true risk R since the distribution \mathcal{Z} is unknown, and an alternative solution is minimization of the empirical risk R_e over the given samples. The generalization error quantifies the gap between the true risk R and the empirical risk R_e , i.e.,

$$\begin{aligned} |R - R_e| &= |\mathbb{E}_{(\mathbf{x}, y) \sim p_{\mathcal{Z}}} [l(f_{\mathbf{w}}(\mathbf{x}), y)] \\ &\quad - \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{S}_l} l(f_{\mathbf{w}}(\mathbf{x}_i), y_i)|. \end{aligned} \quad (11)$$

Thus, the true risk R can be upper-bounded by its empirical error and the generalization error, i.e.,

$$R \leq \underbrace{|R - R_e|}_{\text{generalization error}} + \underbrace{|R_e|}_{\text{empirical error}}. \quad (12)$$

Since the empirical error—namely, the training loss—is close to zero in general, we can consider only the upper bound of the generalization error. Thus, a natural question is whether the coreset selection could provide a tighter generalization bound compared with random sampling for the same pruned training size. To answer this question, we need first to define the empirical error R_r on a sub-training-set \mathcal{S}_r that is generated by random sampling from the full dataset \mathcal{S} as

$$R_r = \frac{1}{|\mathcal{S}_r|} \sum_{\mathbf{x}_i, y_i \in \mathcal{S}_r} l(f_{\mathbf{w}}(\mathbf{x}_i), y_i), \quad (13)$$

where $|\mathcal{S}_r| = N_r$ is the size of \mathcal{S}_r . Hence, the generalization error G_r over \mathcal{S}_r is represented as

$$\begin{aligned} G_r &= |R - R_r| = |\mathbb{E}_{(\mathbf{x}, y) \sim p_{\mathcal{Z}}} [l(f_{\mathbf{w}}(\mathbf{x}), y)] \\ &\quad - \frac{1}{|\mathcal{S}_r|} \sum_{\mathbf{x}_i, y_i \in \mathcal{S}_r} l(f_{\mathbf{w}}(\mathbf{x}_i), y_i)|. \end{aligned} \quad (14)$$

According to previous studies on the generalization analysis of QNNs and quantum kernels [6, 59], with probability at least $1 - \delta$ over \mathcal{S}_r , we have the generalization error of a QNN

$$G_r^{\text{QNN}} \leq \mathcal{O} \left(\sqrt{\frac{m \log(m)}{N_r}} + \sqrt{\frac{\log(1/\delta)}{N_r}} \right), \quad (15)$$

where m is number of the trainable parameters of the QNN, and the generalization error of a quantum kernel

$$G_r^{\text{kernel}} \leq \mathcal{O} \left(\sqrt{\frac{\|\mathbf{w}\|^2}{N_r}} + \sqrt{\frac{\log(4/\delta)}{N_r}} \right). \quad (16)$$

To bound the generalization error G_c on the coreset \mathcal{S}_c , we similarly define the empirical error R_c on the coreset \mathcal{S}_c as

$$R_c = \frac{1}{|\mathcal{S}_c|} \sum_{\mathbf{x}_s, y_s \in \mathcal{S}_c} \gamma_s \cdot l(f_{\mathbf{w}}(\mathbf{x}_s), y_s), \quad (17)$$

where $|\mathcal{S}_c| = N_c$ is the size of \mathcal{S}_c and γ_s is the weight for each coreset example \mathbf{x}_s that is used to make R_c approximate to R_e over the full dataset \mathcal{S} . Then, we can express G_c as

$$G_c = |R - R_c| = |\mathbb{E}_{(\mathbf{x}, y) \sim p_{\mathcal{Z}}} [l(f_{\mathbf{w}}(\mathbf{x}), y)] - \frac{1}{|\mathcal{S}_c|} \sum_{\mathbf{x}_s, y_s \in \mathcal{S}_c} l(f_{\mathbf{w}}(\mathbf{x}_s), y_s)|. \quad (18)$$

On the basis of the triangle inequality, G_c can be bounded as follows:

$$G_c = |R - R_c| \leq \underbrace{|R - R_e|}_{\text{generalization error}} + \underbrace{|R_e - R_c|}_{\text{coreset error}}. \quad (19)$$

Here we name the second term, i.e., $|R_e - R_c|$, the ‘‘coreset error,’’ which describes the gap between the empirical loss on the full dataset and the coreset. As the bound of the generalization-error term, i.e., $|R - R_e|$, is given in previous work [6,59], we focus only on the bound of the coreset error.

By analyzing the coreset error, we first provide the generalization-error bound of QNNs.

Theorem 1 (Generalization-error bound of QNNs on the coreset). Given sample set $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$ with those samples being i.i.d. (independent and identically distributed) drawn from distribution \mathcal{Z} , \mathcal{S}_c is the δ_c cover of \mathcal{S} . Assume that there is a λ_η -Lipschitz-continuous class-specific regression function $\eta(\mathbf{x}) = p(y = c|\mathbf{x})$ and that the loss $l(f_{\mathbf{w}}(\mathbf{x}_s), y_s)$ over the coreset \mathcal{S}_c is zero and bounded by L . We have the following upper bound for the generalization error of QNNs trained on the coreset with

probability $1 - \delta$:

$$G_c^{\text{QNN}} \leq \mathcal{O} \left(\sqrt{\frac{m \log(m)}{N_t}} + \sqrt{\frac{\log(1/\delta)}{N_t}} + \delta_c (\lambda_\eta L n_c + d \sqrt{d_{\mathbf{x}}} \max_j |\mathbf{w}_j| |M| (|M| + \max |y|)) \right), \quad (20)$$

where m is the number of parameters in the QNN, d is the number of QNN layers, n_c is the number of classes, M is the measurement operator, and $d_{\mathbf{x}}$ is the feature dimension.

For the coreset error $|R_e - R_c|$, we assume that the training error on the coreset is equal to zero, and thus it becomes the average error that can be bounded with radius δ_c over the entire dataset determined by the k -center covering problem shown in Fig. 1, which is related to the data-pruning rate $\zeta = |\mathcal{S}_c|/|\mathcal{S}|$. Combining the generalization bound on the full dataset and the bound of the risk gap between the full dataset and the coreset, we have that the generalization error of the QNN on the coreset is bounded mainly by two terms, i.e., $\mathcal{O}(\sqrt{m \log(m)/N_t} + \delta_c)$. Thus, G_c^{QNN} will give a tighter bound than G_r^{QNN} when we carefully choose the data-pruning rate ζ . It is clear that G_c^{QNN} gives a tighter bound on the first term since $N_r < N_t$. For the second term, as δ_c is related to ζ , a low ζ will cause δ_c to become large, leading to a high approximation error. Conversely, a high ζ will decrease the approximation error, but the acceleration of training will disappear because N_c is approximately equal to N_t .

We next provide the generalization-error bound of quantum kernels on the coreset.

Theorem 2 (Generalization-error bound of quantum kernels on the coreset). Given sample set $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$ with those samples being i.i.d. (independent and identically distributed) drawn from distribution \mathcal{Z} , \mathcal{S}_c is the δ_c cover of \mathcal{S} . Assume that there is a λ_η -Lipschitz-continuous class-specific regression function $\eta(\mathbf{x}) = p(y = c|\mathbf{x})$ and that the loss $l(f_{\mathbf{w}}(\mathbf{x}_s), y_s)$ over the coreset \mathcal{S}_c is zero and bounded by L . We have the following upper bound for the generalization error of the SVM with a quantum kernel trained on the coreset with probability $1 - \delta$:

$$G_c^{\text{kernel}} \leq \mathcal{O} \left(\sqrt{\frac{\|\mathbf{w}\|^2}{N_t}} + \sqrt{\frac{\log(4/\delta)}{N_t}} + \delta_c (\lambda_\eta L n_c + N_c \sqrt{d_{\mathbf{x}}} \max_j |\mathbf{w}_j| \cdot (1 + (N_q - 1)r)) \right), \quad (21)$$

where n_c is the number of classes, $d_{\mathbf{x}}$ is the feature dimension of \mathbf{x} , N_q is the size of the mapped quantum state $|\mathbf{x}\rangle$, and r is the maximum value of the feature \mathbf{x} .

Similarly to the analysis of the generalization error of QNNs on the coreset, the generalization error of an SVM with a quantum kernel is also mainly bounded by two terms, i.e., $\mathcal{O}(\sqrt{(\|\mathbf{w}\|^2)/N_t} + \delta_c)$, which indicates we have results similar to those for G_c^{QNN} . Here, when we use coreset selection to reduce the size of the training examples, we reduce the complexity of getting the kernel matrix, which provides $\mathcal{O}(k^2)$ speedup, where $k = N_t/N_c$, while also having a provable generalization guarantee.

It is not easy to propose a universal trick for selecting an appropriate data-pruning rate ζ that determines δ_c to achieve good generalization performance because it is related to the distribution of the given training set \mathcal{S} and the unknown true data distribution \mathcal{Z} . Nevertheless, we report some numerical experiments on various data and models; the results not only support the analytical findings but also might provide some practical advice for the selection of ζ . Consequently, if we carefully choose the data-pruning rate and the size of the training set to scale at least quasi-linearly with the number of gates used, we are able to accelerate the quantum machine learning model with a performance guarantee. These results provide effective and practical guidance for achieving accurate and reliable performance with a reasonable model complexity and sample complexity.

VI. NUMERICAL RESULTS

In this section, we report extensive numerical simulations to explore the performance of the proposed coreset-selection method. Specifically, we use our proposal to accomplish three learning tasks, which are synthetic data classification, quantum correlation identification, and quantum compiling.

A. Synthetic data classification by quantum kernels

We first use the quantum kernel to classify a synthetic dataset with a coreset. The construction rule for the synthetic dataset mainly follows the method in Ref. [6].

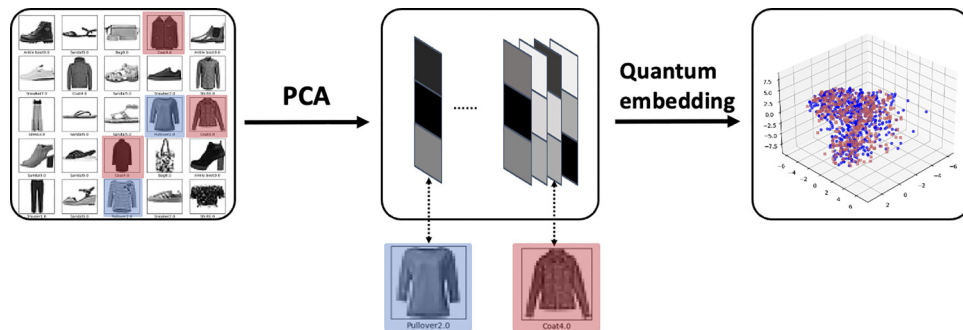


FIG. 4. The synthetic dataset used adapted from Fashion-MINIST [78]. We use principal-component analysis (PCA) to get the low-dimensional representation, and then embed the reduced data into the quantum Hilbert space. In the end, we relabel the data according to maximization of the geometric difference between classical and quantum kernels in Eq. (22).

Specifically, given samples $\{\mathbf{x}_i, y_i\}_{i=1}^N$ that are independently sampled from the distribution \mathcal{X} , the corresponding labels are modified according to the maximized geometric difference given by

$$\max_{y \in \mathbb{R}^N} \frac{\sum_{i=1}^N \sum_{j=1}^N (K^Q)_{ij}^{-1} y_i y_j}{\sum_{i=1}^N \sum_{j=1}^N (K^C)_{ij}^{-1} y_i y_j}, \quad (22)$$

where K^Q and K^C denote the quantum kernel and the classical kernel, respectively. The optimal solution of Eq. (22) yields the modified labels \mathbf{y}^* such that the geometric difference is maximized:

$$\mathbf{y}^* = \text{sign}(\sqrt{K^Q} \mathbf{v}), \quad (23)$$

where \mathbf{v} is the eigenvector of $\sqrt{K^Q}(K^C)^{-1}\sqrt{K^Q}$ with the maximum eigenvalue, and $\text{sign}(\mathbf{z})$ is the elementwise function such that set the i th element as $+1$ if $z_i > \text{median}(\mathbf{z})$ otherwise set as -1 . As proved in Ref. [6], quantum kernels can achieve quantum advantages when learning this dataset.

An illustration of the synthetic dataset construction is shown in Fig. 4. Concretely, in our numerical simulations, the synthetic dataset is based on Fashion-MNIST [78]. As the dimension of the vectorized data of Fashion-MNIST is too high for a noisy intermediate-scale quantum (NISQ) device, we preprocess the data as the low-dimensional representation by principal-component analysis and then relabel the class of each data point according to Eq. (22). Besides, the element of the classical kernel K_{ij}^C is given by the radial-basis-function kernel $K_{ij}^C = \exp(-(\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2))$. The quantum kernel K_{ij}^Q is generated through our encoding the data points into the N_q -qubit Hilbert space by a quantum circuit U_e ,

$$K_{ij}^Q = \text{tr}(\rho(\mathbf{x}_i)\rho(\mathbf{x}_j)), \quad (24)$$

where $\rho(\mathbf{x}) = U_e(\mathbf{x})|0\rangle\langle 0|U_e^\dagger(\mathbf{x})$. We can further assume that the following form of $U_{\mathbf{x}}$ is implemented through

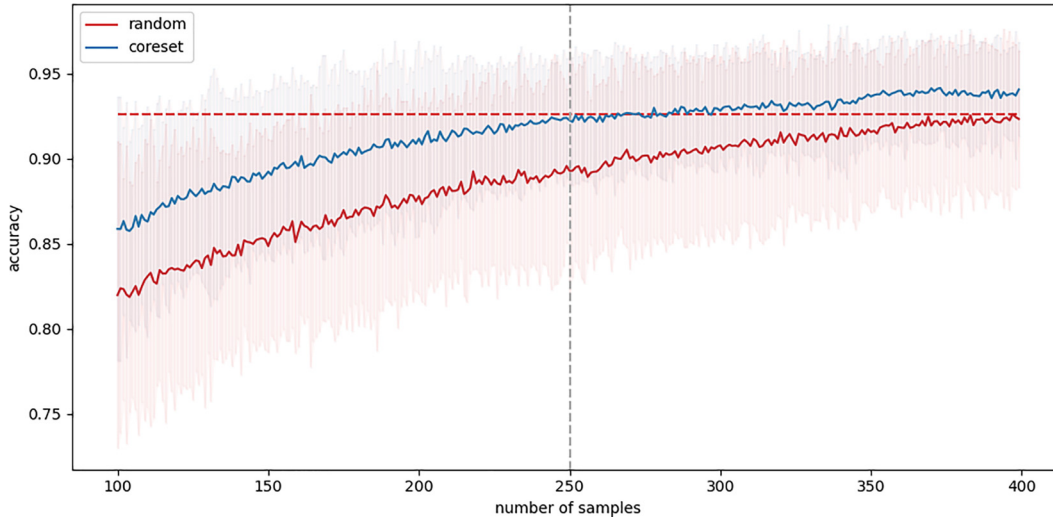


FIG. 5. Performance of the proposed model assessed by our comparing random sampling and coreset selection in terms of their classification abilities on synthetic data. The solid line represents the average test accuracy of models trained with these methods. The shaded area refers to the range of the test accuracy. The dotted red line denotes the maximum accuracy achieved by random sampling over 400 examples.

quantum gates in an N_q -qubit circuit: $U_{\mathbf{x}} = (U(\mathbf{x})H^{\otimes N_q})^2 |0\rangle^{\otimes N_q}$, where $U(\mathbf{x}) = \exp\left(\sum_{j=1}^{N_q} \mathbf{x}_j \sigma_j^Z + \sum_{j,j'=1}^{N_q} \mathbf{x}_j \mathbf{x}_{j'} \sigma_j^Z \sigma_{j'}^Z\right)$.

Once the synthetic dataset had been prepared, we conducted experiments on training sets of various sizes and subsequently tested on 200 unseen examples to get the test accuracy. Instead of independently and randomly choosing the training data from the entire set \mathcal{S} , we first solved the k -center problem over the set \mathcal{S} with 1000 examples that are equivalent to form the coreset \mathcal{S}_c . In Fig. 5, we show a comparison of the classification performance under these settings. Figure 5 depicts the correlation between the size of the training set, obtained with random sampling and coreset selection, and the corresponding test accuracy. For experiments involving the same number of training examples, we conducted five independent trials, each using randomized initialization, resulting in the shaded area shown in Fig. 5. The number of samples selected in the proposed coreset method refers to the parameter k in the k -center problem, where we vary it from 100 to 400 to estimate its impact on model performance. As k increases, the performance improves due to the model being trained on a larger number of training samples. The broader dataset encompasses more information about the true data distribution, leading to better performance. The average test performance of these trials is plotted as a solid line. We also highlight the best test accuracy of random sampling as the dashed red line. From the results shown in Fig. 5, for training with the same number of samples, the coreset method exhibits higher test accuracy. Meanwhile, to achieve competitive performance, the coreset method requires nearly half of the samples. For instance,

even though the test accuracy of the coreset over 250 samples is around 0.93, which is slightly higher than the 0.88 of random sampling, it has already achieved competitive performance of the model over 400 random sampling data. The findings support our analytical results: QNNs trained on the coreset have better generalization performance than those with randomly picked training data under appropriate k . The coreset-enhanced classifier significantly increases the training efficiency, achieving competitive performance while using only approximately 50% of the training examples compared with random sampling.

B. Correlation identification by QNNs

Nonclassical correlation plays a core role in quantum information and quantum computation [79]. Nevertheless, identifying the nonclassical correlation of a given quantum state is a challenging task. Yang *et al.* [80] explored classifying nonclassical correlation experimentally with machine-learning techniques. Consider the family of quantum states characterized by p and θ with the following form:

$$\rho_{AB}(p, \theta) = p|\psi_\theta\rangle\langle\psi_\theta| + (1-p)\frac{\mathbb{I}}{2} \otimes \text{tr}_A(|\psi_\theta\rangle\langle\psi_\theta|), \quad (25)$$

where $p \in (0, 1)$, $\theta \in (0, 2\pi)$, and state $|\psi_\theta\rangle = \cos(\theta)|00\rangle + \sin(\theta)|11\rangle$. There are rules to determine the nonclassical correlation of quantum states ρ_{AB} , including separable, entangled, one-way-steerable, and nonlocal states:

- (1) According to the Peres–Horodecki criterion, the states are *separable* when $p < \frac{1}{3}$, otherwise they are entangled.

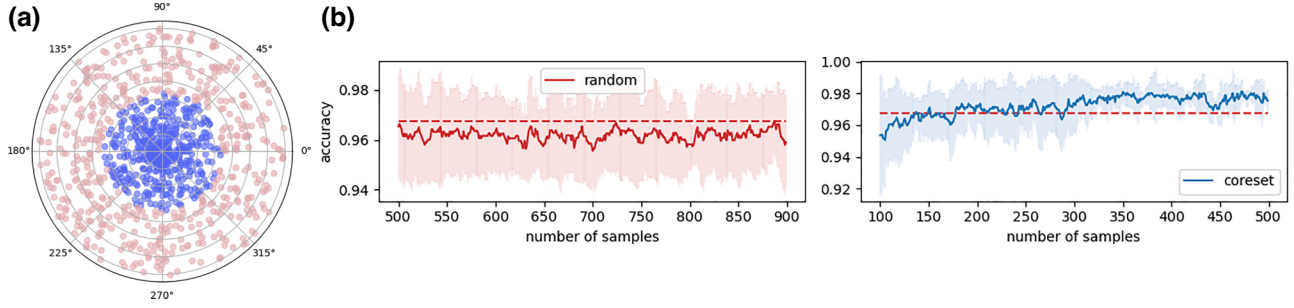


FIG. 6. Results related to correlation identification. (a) The quantum states are represented as dots in a polar plot. The radius of the polar plot represents the parameter p , which varies from 0 to 1. The phase stands for the parameter θ , which varies from 0 to 2π . The blue and orange dots indicate the separable and entangled states, respectively. (b),(c) Comparison of test accuracy between the classifier with (b) random sampling and (c) coreset selection for different sizes of the training set.

(2) When $(1/\sqrt{2}) < p < (1/(\sqrt{1 + \sin^2(2\theta)}))$, the quantum state is *one-way steerable*.

(3) When $p > (1/(\sqrt{1 + \sin^2(2\theta)}))$, the state is *nonlocal*.

Therefore, to identify the nonlocal correlation of a given state under the learning framework, we can label the quantum states with different nonclassical correlations according to the above criteria and create the dataset $\{\rho_j^{AB}, y_j\}_{j=1}^N$, where y_i represents the type of correlation, i.e., separable, entangled, one-way steerable, and nonlocal.

To further enhance the methods, we apply coreset selection to this learning task, which reduces the number of training samples and shortens the runtime. For classifying the quantum correlation, we uniformly pick the parameters $p \in (0, 1)$ and $\theta \in (0, 2\pi)$ to generate the 1000 quantum states as the full training set as shown in Fig. 6(a). Then we label the class of correlation in terms of the criteria listed above. Here we continue with the same experimental setup as in the previous section, where we examine the test accuracy of the model, trained over various sample sizes, on 200 unseen random samples.

The dashed red lines in Figs. 6(b) and 6(c) denote the maximum test accuracy achieved with the classifier through random sampling, with a maximum of 900 training samples. When the coreset method is used to prune the dataset, for sample sizes exceeding 180, the average test accuracy obtained is almost higher than the greatest accuracy achieved by random sampling. We emphasize that to achieve competitive performance, the coreset selection requires a far smaller data size than random sampling. This indicates that coreset selection could provide better performance than training on random sampling data, which matches our theoretical findings.

C. Quantum compiling by QNNs

Compiling a unitary into a sequence of gates is a challenging and high-profile task for NISQ devices. With the

limitation of current NISQ hardware, compiling a unitary should not only take account of the function but should also consider the connectivity and depth of the output circuit. Recently, various methods for quantum compiling have been proposed under the framework of variational quantum algorithms [49,59,81]. In general, these algorithms consider the compiling task as an optimization problem on a given compact quantum gate set that consists of fixed and parametrized quantum gates. The goal is to optimize the structure and gate parameters such that the proposed quantum circuit approximates the given unitary.

Here we consider a method that tackles the compiling task by a quantum machine learning protocol. Given an n -qubit target unitary U , the training data consist of random input states and their corresponding output when U applied on input states $|\psi_j$, i.e., $\{|\psi_j\rangle, U|\psi_j\rangle\}_{j=1}^N$. To approximate the target unitary U , one can simply minimize the empirical loss of the squared trace distance between target states $U|\psi_j\rangle$ and parametrized output states $V(\theta)|\psi_j\rangle$ and over randomly sampled states in Hilbert space. Concretely, we randomly pick the quantum gates from the gate pool consisting of single parametrized gates $\{R_x, R_y, R_z\}$ and a controlled-NOT gate to build the target quantum circuits U . Then we set the input as the random state $|\psi_i\rangle$ in Hilbert space and get training pairs $\{|\psi_j\rangle, U|\psi_j\rangle\}_{j=1}^{1000}$. In Fig. 7(b), we present a comparison between the models trained on random samples and the coreset with various data sizes and pruning ratios. We separately estimate the performance of the model with $\zeta = \{0.8, 0.4, 0.6, 0.2\}$. Each data point with a different color in the plot corresponds to a different value of ζ . For the compiling task on a specific unitary, we found that 1000 training examples are redundant, and the effective size of the training data scales linearly with the system size, which is similar to what previous work found [59]. Twenty examples are sufficient for training the variational compiler to achieve a reasonable performance for a six-qubit system. Since the randomly sampled training points are likely to be uniformly located in Hilbert space and the proposed coreset-selection

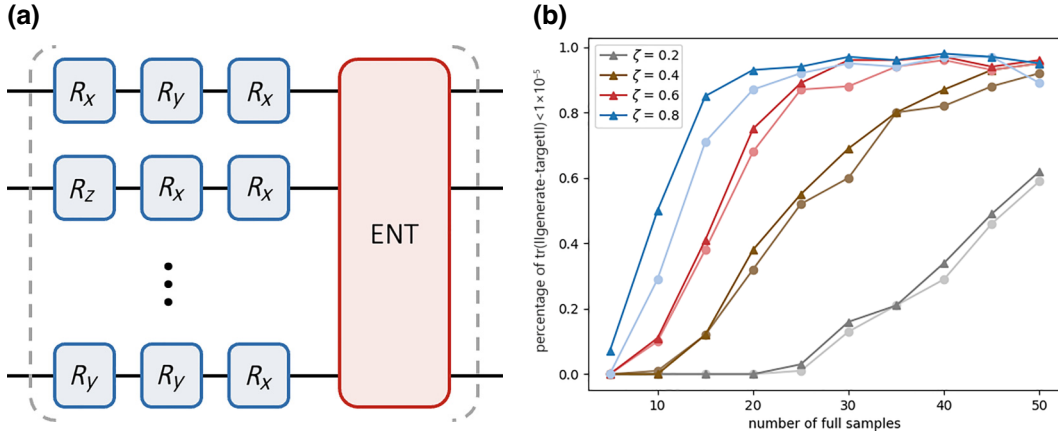


FIG. 7. Results of unitary compiling. (a) The target unitary is used in numerical simulations. (b) Performance comparison between the different pruning ratios ζ of the coreset on the compiling task. The solid dark and light lines represent the models trained on the coreset and random samples. The vertical axis represents the percentage of states in the test set for which the trace distance to their corresponding targets is below 10^{-5} . ENT, .

approach also uses k centers to uniformly cover the entire set, the compilers trained on the coreset with different values of ζ have performance similar to that of those trained on the randomly sampled set. Moreover, we highlighted that the model's inferior performance compared with other cases, when the pruning ratio $\zeta = 0.2$, is due to the inadequate training data, which hinders the development of good generalization behavior. This also suggests that while the coreset method can reduce the sample size effectively, inadequate data still impact the model's performance. Additionally, this phenomenon is intertwined with the geometric attributes of data distributions, which may result in differing compression-ratio thresholds.

VII. DISCUSSION

In this work, we investigate enhancing the QML model from the data-engineering perspective and attempt to alleviate a practical problem when handling a large volume of data samples. In particular, we consider the coreset construction as a k -set cover problem and then analyze the generalization performance of QML models on the coreset. Our investigation of various learning scenarios, including on the classification of synthetic data, identification of nonclassical correlations in quantum states, and quantum circuit compilation, confirms the extreme improvements in the effectiveness of our proposal.

Our research findings highlight the considerable enhancement in model training achieved through the use of the coreset method. Data pruning contributes to increased training efficiency. Besides, it also helps filter out noisy data, thereby enhancing model performance. It is evident that selecting a sparser coreset enforces a more-rigorous upper bound on the number of trainable gates and an appropriate data-pruning rate. The size of the training set should scale at least quasilinearly with the number of gates.

These findings provide effective and practical guidelines to achieve accurate and reliable results with a reasonable configuration of gates and training data. Although we have increased model-training efficiency through data-pruning methods, there is still significant room for enhancement. For instance, without prior assumptions about the dataset, identifying the ideal k value or pruning ratio ζ becomes even more complex. One way is to leverage the geometric properties of the dataset, which might offer a more-efficient method for identifying k or the pruning ratio ζ . Besides, one can also improve our proposal by introducing the influencing functions that might better characterize the impact of data variations on the model, thus achieving more-precise data filtering. We leave this to future work. We hope that the work presented in this article will offer valuable insights and guidance for future practical research in quantum machine learning, from theoretical or practical aspects, on NISQ devices.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No. U2330201). The numerical experiments in this work were supported by the High-Performance Computing Platform of Peking University.

APPENDIX

1. Proof of Theorem 1

Theorem 3 (Generalization bound for QNNs [59]). Let the QNN to be trained consist of m trainable parametrized two-qubit or one-qubit gates, and an arbitrary number of nontrainable fixed gates. Suppose that, given training data $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$ and a loss function $l(f_{\theta}; \mathbf{x}_i, y_i)$ bounded by L , our optimization yields the optimized parameters θ^* .

Then, according to Theorem C.6 in Ref. [59], the generalization error of the optimized QNN is bounded by the following form with probability at least $1 - \delta$ over the choice of independent and identically distributed training data \mathcal{S}_t from \mathcal{Z} :

$$|R - R_e| < \mathcal{O} \left(\sqrt{\frac{m \log(m)}{N_t}} + \sqrt{\frac{\log(1/\delta)}{N_t}} \right). \quad (\text{A1})$$

Lemma 1. The loss function $l(f_\theta(\mathbf{x}), y)$ of the d -layer QNN introduced in Sec. III C is the λ -Lipschitz in the feature space with

$$\lambda = 2d\sqrt{d_{\mathbf{x}}} \max_j |\mathbf{w}_j| |M| (|M| + \max |y|). \quad (\text{A2})$$

Proof. Assume that we have given data $\mathbf{x} = (x_1, \dots, x_{d_{\mathbf{x}}})$ and that data-encoding scheme used by the QNN is the d -layer reuploading scheme. Then the mapped quantum state is written as

$$|\mathbf{x}, \theta\rangle = U(\mathbf{x}, \theta^{(d)}, \mathbf{w}^{(d)}) \dots U(\mathbf{x}, \theta^{(1)}, \mathbf{w}^{(1)}) |0\rangle, \quad (\text{A3})$$

where \mathbf{w} is the encoding parameters in data reuploading, and the i th layer $U(\mathbf{x}, \theta^{(i)}, \mathbf{w}^{(i)})$ can be represented as

$$U(\mathbf{x}, \theta^{(i)}, \mathbf{w}^{(i)}) = U(\theta^{(i)}) \prod_k \exp(iw_k^{(i)} x_k P_k), \quad (\text{A4})$$

where P is a Pauli gate, and θ and \mathbf{w} are both trainable parameters.

Assume that the Jacobian vector J of the loss function $l = (\langle \mathbf{x}, \theta | M | \mathbf{x}, \theta \rangle - y)^2$ is bounded in the feature space. We now investigate the Lipschitz constant of l :

$$|l(\mathbf{x}) - l(\mathbf{x}')| \leq \max \|J\|_2 \|\mathbf{x} - \mathbf{x}'\|_2 \text{ for all } \mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_{\mathbf{x}}}, \quad (\text{A5})$$

where the Jacobian matrix is given by

$$J = \left[\frac{\partial l}{\partial x_1}, \dots, \frac{\partial l}{\partial x_{d_{\mathbf{x}}}} \right]. \quad (\text{A6})$$

Thus, the maximum norm of the Jacobian matrix J can be bounded by the maximum norm of the derivative of l with respect to \mathbf{x} :

$$\max \|J\|_2 \leq \sqrt{d_{\mathbf{x}}} \max_j \left| \frac{\partial l}{\partial x_j} \right|, \quad (\text{A7})$$

where $|\partial l / \partial x_j|$ can be bounded by

$$\begin{aligned} \left| \frac{\partial l}{\partial x_j} \right| &= \left| (\langle \psi | M | \psi \rangle - y) \cdot 2\mathcal{R} \left(\langle \psi | M \frac{\partial |\psi\rangle}{\partial x_j} \right) \right| \\ &\leq (|M| + \max |y|) \cdot 2|M| \cdot \left| \frac{\partial |\psi\rangle}{\partial x_j} \right|_2, \end{aligned} \quad (\text{A8})$$

where $|M|$ denotes the spectral norm of the measurement operator. To bound $\|(\partial |\psi\rangle / \partial x_j)\|_2$, let $U^{(k)} =$

$U(\mathbf{x}, \theta^{(k)}, \mathbf{w}^{(k)})$. We first apply the product rule to $(\partial |\psi\rangle / \partial x_j)$:

$$\frac{\partial |\psi\rangle}{\partial x_j} = \sum_{k=1}^d U^{(d)} \dots i w_j^{(k)} P_j U^{(k)} \dots U^{(1)} |0\rangle. \quad (\text{A9})$$

As the norm of the operators $U^{(k)}$ and P_j is equal to 1, we can bound $\|(\partial |\psi\rangle / \partial x_j)\|_2$ by

$$\begin{aligned} \left| \frac{\partial |\psi\rangle}{\partial x_j} \right|_2 &\leq \sum_{k=1}^d \left| U^{(d)} \cdot w_j^{(k)} P_j U^{(k)} \cdot U^{(1)} |0\rangle \right|_2 \\ &= \sum_{k=1}^d |w_j^{(k)}| \leq d \max_j |\mathbf{w}_j|. \end{aligned} \quad (\text{A10})$$

Hence, we can bound $|\partial l / \partial x_j|$ by

$$\left| \frac{\partial l}{\partial x_j} \right| \leq 2d \max_j |\mathbf{w}_j| \cdot |M| \cdot (|M| + \max |y|). \quad (\text{A11})$$

Provided the feature space and the label space are bounded, the loss function is λ Lipschitz continuous with

$$\lambda = 2d\sqrt{d_{\mathbf{x}}} \max_j |\mathbf{w}_j| \cdot |M| (|M| + \max |y|). \quad (\text{A12})$$

■

We ignore the residual training error in the coreset from now on. For the n_c -class classification problem, we assume that there are class-specific regression functions that represent the probability of a class label in the neighborhood of a feature vector $\eta_c(\mathbf{x}) = p(y = c | \mathbf{x})$ and that they satisfy the property of being λ_η Lipschitz continuous with respect to the feature space of \mathbf{x} . Then we present Lipschitz constants for the loss functions of a QNN and derive the upper bound for the generalization error.

Proof of Theorem 1. By the triangle inequality,

$$G_c^{\text{QNN}} = |R - R_c^{\text{QNN}}| \leq |R - R_e^{\text{QNN}}| + |R_e^{\text{QNN}} - R_c^{\text{QNN}}|. \quad (\text{A13})$$

Since $|R - R_e^{\text{QNN}}|$ is bounded by Theorem 3,

$$|R - R_e^{\text{QNN}}| < \mathcal{O} \left(\sqrt{\frac{m \log(m)}{N_t}} + \sqrt{\frac{\log(1/\delta)}{N_t}} \right). \quad (\text{A14})$$

Let us attempt to relate R_e to the coreset. Since we assume that the training loss over the coreset R_c^{QNN} is equal to zero, only R_e^{QNN} should be considered. Here we allow differently labeled samples to appear mixed in a local region of the

feature space. Instead of regarding the probability of each label jumping between 0 and 1, we use the local class-specific regression function η to describe the probability of each label close to a point:

$$\begin{aligned} R_e^{\text{QNN}} &= \frac{1}{|\mathcal{X}|} \sum_{\mathbf{x}_j \in \mathcal{X}} \sum_{c_i} (\eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_j), c_i)) \\ &= \mathbb{E}_{\mathbf{x}_j \in \mathcal{X}} \sum_{c_i} (\eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_j), c_i)). \end{aligned} \quad (\text{A15})$$

For each (\mathbf{x}_j, y_j) drawn from the training set, we are guaranteed that the distance between any drawn sample (\mathbf{x}_c, y_j) and its nearest center (\mathbf{x}_c, y_j) in coreset with the same class is less than δ_c . If locally the class-specific regression function $\eta_{c_i} \neq 0$ for another label c_i , with $c_i \neq y_j$, then we would have at least one differently labeled sample within δ_c from \mathbf{x}_c and $2\delta_c$ from \mathbf{x}_j . This other-class sample in the same circle, in turn, guarantees an other-class center $\mathbf{x}_c(c_i)$ within $3\delta_c$ from \mathbf{x}_j . For each \mathbf{x}_j , we have

$$\begin{aligned} &\sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_j), c_i) \\ &= \sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot [l(f_\theta(\mathbf{x}_j), c_i) - l(f_\theta(\mathbf{x}_j(c_i)), c_i)] \\ &\quad + \sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_c(c_i)), c_i) \\ &\quad + \sum_{c_i} \eta_{c_i}(\mathbf{x}_t) \cdot l(f_\theta; \mathbf{x}_c(c_i), c_i) \\ &= \sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot [l(f_\theta(\mathbf{x}_j), c_i) - l(f_\theta(\mathbf{x}_j(c_i)), c_i)] \\ &\quad + \sum_{c_i} [\eta_{c_i}(\mathbf{x}_j) - \eta_{c_i}(\mathbf{x}_c(c_i))] \cdot l(f_\theta(\mathbf{x}_c(c_i)), c_i) \\ &\quad + \sum_{c_i} \eta_{c_i}(\mathbf{x}_c(c_i)) \cdot l(f_\theta(\mathbf{x}_c(c_i)), c_i). \end{aligned} \quad (\text{A16})$$

The last term is the loss of the center weighted by the local probability of being of the same class as that center. For a specific center, it will be called by both same-class samples and other-class samples that find it closest in the coreset, and the number of calls is approximately proportional to the number of samples covered by the center. With the factor $\eta_{c_i}(\mathbf{x}_c(c_i))$ in each call, the corresponding weight for

a center in the coreset should be the number of samples covered times the local probability of being of the center class, i.e., the number of same-class samples covered by the center. This justifies our choice of the weight in Eq. (17). Assuming the weighted loss is trained to zero, the form of the previous expression is then simplified to

$$\begin{aligned} &\sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_j), c_i) \\ &= \sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \cdot [l(f_\theta(\mathbf{x}_j), c_i) - l(f_\theta(\mathbf{x}_c(c_i)), c_i)] \\ &\quad + \sum_{c_i} [\eta_{c_i}(\mathbf{x}_j) - \eta_{c_i}(\mathbf{x}_c(c_i))] \cdot l(f_\theta(\mathbf{x}_c(c_i)), c_i). \end{aligned} \quad (\text{A17})$$

The two parts in the last equation can be separately bounded by the Lipschitz continuity λ_ζ and λ_l of η_{c_i} and l , and we assume that l is bounded by L . Substituting the corresponding constants, we have

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}_j \in \mathcal{X}} \sum_{c_i} (\eta_{c_i}(\mathbf{x}_j) \cdot l(f_\theta(\mathbf{x}_j), c_i)) \\ &\leq \left(\sum_{c_i \neq y_j} \eta_{c_i}(\mathbf{x}_j) \cdot 3\delta_c + \eta_{y_j}(\mathbf{x}_j) \cdot \delta_c \right) \lambda_l \\ &\quad + ((n_c - 1) \cdot 3\delta_c + \delta_c) L \lambda_\eta \\ &\leq \left(\sum_{c_i} \eta_{c_i}(\mathbf{x}_j) \right) 3\delta_c \lambda_l + (3n_c - 2)\delta_c \lambda_\eta L \\ &= 3\delta_c \lambda_l + (3n_c - 2)\delta_c \lambda_\eta L. \end{aligned} \quad (\text{A18})$$

According to Hoeffding's bound and the assumption of zero coreset training loss, we have the coreset error of the QNN with probability $1 - \delta$:

$$\begin{aligned} |R_e^{\text{QNN}} - R_c^{\text{QNN}}| &\leq 3\delta_c \lambda_l + (3n_c - 2)\delta_c \lambda_\eta L \\ &\quad + \sqrt{\frac{L^2 \log(1/\delta)}{2n}}. \end{aligned} \quad (\text{A19})$$

Combing the bound of $|R - R_e^{\text{QNN}}|$, we have the generalization bound of the QNN on the coreset with probability $1 - \delta$:

$$G_c^{\text{QNN}} \leq \mathcal{O} \left(\sqrt{\frac{m \log(m)}{N_t}} + \sqrt{\frac{\log(1/\delta)}{N_t}} + \delta_c (\lambda_\eta L n_c + d \sqrt{d_{\mathbf{x}}} \max_j |\mathbf{w}_j| |M| (|M| + \max |y|)) \right). \quad (\text{A20})$$

■

2. Proof of Theorem 2

Theorem 4 (Generalization bound for learning with quantum kernels [13]). Suppose the parameters \mathbf{w} of the hypothesis $f_{\mathbf{w}}$ are optimized to be \mathbf{w}^* by our minimizing the loss $l(f_{\mathbf{w}}(\mathbf{x}), y) = |\min(1, \max(-1, \sum_j w_j \kappa(\mathbf{x}_j, \mathbf{x}))) - y|$ on a training set $\{\mathbf{x}_i, y_i\}_{i=1}^{N_t}$. Then, according to Theorem 2 in [6], the generalization error of the optimized hypothesis is bounded by the following form with probability at least $1 - \delta$ over the choice of independent and identically distributed training data \mathcal{S} from \mathcal{Z} :

$$|R - R_e^{\text{kernel}}| < \mathcal{O} \left(\sqrt{\frac{\lceil \|\mathbf{w}\|^2 \rceil}{N_t}} + \sqrt{\frac{\log(4/\delta)}{N_t}} \right). \quad (\text{A21})$$

Lemma 2. The function $l(f_{\mathbf{w}}(\mathbf{x}), y) = |\min(1, \max(-1, \sum_j w_j \kappa(\mathbf{x}_j, \mathbf{x}))) - y|$, where $\kappa(\mathbf{x}, \mathbf{x}')$ is the quantum kernel function, is λ Lipschitz continuous in the feature space, with

$$\lambda = 2k_c \sqrt{d_{\mathbf{x}}} \max_i |w_i| \cdot (1 + (N_q - 1)r). \quad (\text{A22})$$

Proof. This follows immediately from evaluation of the norm of the weight vector in the mapped Hilbert space. The function l can be viewed as $l = g \circ f$, where

$$g(f) = |\min(1, \max(-1, f)) - y| \quad (\text{A23})$$

and

$$f(\mathbf{w}, \mathbf{x}) = \sum_{j=1}^{k_c} w_j \kappa(\mathbf{x}_j, \mathbf{x}), \quad (\text{A24})$$

where $\kappa : \mathbb{R} \cdot \mathbb{R} \rightarrow \mathbb{R}$ is the quantum kernel function and k_c is the size of the coreset. The function g has a Lipschitz constant of 1 with respect to f . Concretely, it is defined as follows:

$$\kappa(\mathbf{x}_j, \mathbf{x}_k) = \langle \phi(\mathbf{x}_j), \phi(\mathbf{x}_k) \rangle = |\langle 0 | U_{\mathbf{x}_j}^\dagger U_{\mathbf{x}_k} | 0 \rangle|^2, \quad (\text{A25})$$

where $U_{\mathbf{x}} = (U(\mathbf{x})H^{\otimes N_q})^2 | 0 \rangle^{\otimes N_q}$, $U(\mathbf{x}) = \exp\left(\sum_{j=1}^{N_q} \mathbf{x}_j \sigma_j^Z + \sum_{j,j'=1}^{N_q} \mathbf{x}_j \mathbf{x}_{j'} \sigma_j^Z \sigma_{j'}^Z\right)$, and N_q is the system size of the mapped state.

For any fixed \mathbf{w} , we have

$$\begin{aligned} |f(\mathbf{w}, \mathbf{x}) - f(\mathbf{w}, \mathbf{x}')| &\leq \max_{\mathbf{x}} \sqrt{\sum_{j=1}^{d_{\mathbf{x}}} \left| \frac{\partial f}{\partial x_j} \right|^2} \cdot |\mathbf{x} - \mathbf{x}'| \\ &\leq \sqrt{d_{\mathbf{x}}} \max_j \left| \frac{\partial f}{\partial x_j} \right| \cdot |\mathbf{x} - \mathbf{x}'|. \end{aligned} \quad (\text{A26})$$

Assume that we train the SVM with a quantum kernel on a coreset $\{\mathbf{x}_i\}_{i=1}^{k_c}$, and let $\rho(\mathbf{x}) = U_{\mathbf{x}} | 0 \rangle \langle 0 | U_{\mathbf{x}}^\dagger$. Then we have

$$\begin{aligned} \left| \frac{\partial f}{\partial x_j} \right| &= 2 \left| \sum_{k=1}^{k_c} w_k \mathcal{R} \left(\langle 0 | \frac{\partial U^\dagger(\mathbf{x})}{\partial x_j} \rho(\mathbf{x}_k) U(\mathbf{x}) | 0 \rangle \right) \right| \\ &\leq 2k_c \cdot \max_k |\mathbf{w}_k| \cdot \left| \langle 0 | \frac{\partial U^\dagger(\mathbf{x})}{\partial x_j} \rho(\mathbf{x}_k) U(\mathbf{x}) | 0 \rangle \right| \\ &= 2k_c \cdot \max_k |\mathbf{w}_k| \cdot \left| \frac{\partial U(\mathbf{x})}{\partial x_j} \right|_2. \end{aligned} \quad (\text{A27})$$

Since $U(\mathbf{x}) = \exp(\sum_{j=1}^{N_q} \mathbf{x}_j \sigma_j^Z + \sum_{j,j'=1}^{N_q} \mathbf{x}_j \mathbf{x}_{j'} \sigma_j^Z \sigma_{j'}^Z)$, we denote $U_s = \prod_{k=1}^{N_q} \exp(ix_k \sigma_k^Z)$ and $U_e = \prod_{k,l=1}^{N_q} U_{k,l}$, where $U_{k,l} = \exp(ix_k x_l \sigma_k^Z \sigma_l^Z)$. Then the norm of the derivative of $U(\mathbf{x})$ with respect to x_j can be bounded:

$$\left| \frac{\partial U(\mathbf{x})}{\partial x_j} \right|_2 = \left| \frac{\partial U_s}{\partial x_j} U_e + U_s \frac{\partial U_e}{\partial x_j} \right| \quad (\text{A28})$$

$$\leq \left| \frac{\partial U_s}{\partial x_j} \right|_2 \cdot |U_e|_2 + |U_s|_2 \cdot \left| \frac{\partial U_e}{\partial x_j} \right|_2 \quad (\text{A29})$$

$$\begin{aligned} &\leq |i\sigma_j^Z| \cdot \left| \exp(ix_j \sigma_j^Z) \right| \cdot \left| \prod_{k \neq j} \exp(ix_k \sigma_k^Z) \right| \\ &\quad + \left| \sum_{k \neq j}^{N_q} ix_k \sigma_k^Z \sigma_j^Z U_{k,j} \cdot \prod_{m < n, (m,n) \neq (k,j) \text{ or } (j,k)}^{N_q} U_{m,n} \right| \end{aligned} \quad (\text{A30})$$

$$\leq 1 + (N_q - 1) \max_j |x_j|. \quad (\text{A31})$$

If we substitute Eq. (A31) into Eq. (A27), we have

$$\left| \frac{\partial f}{\partial x_j} \right| \leq 2k_c \max_i |w_i| \cdot (1 + (N_q - 1)r). \quad (\text{A32})$$

Assuming r , the maximum feature value of \mathbf{x} , is bounded and if we substitute Eq. (A32) into Eq. (A26), we can conclude that the function $l(f_{\mathbf{w}}(\mathbf{x}), y)$ is $2k_c \sqrt{d_{\mathbf{x}}} \max_i |w_i| \cdot (1 + (N_q - 1)r)$ Lipschitz continuous. ■

Proof of Theorem 2. The proof of Theorem 2 is similar to the proof of Theorem 1, but with replacement of the relevant Lipschitz constant and generalization bound from Lemma 2 and Theorem 1, respectively. ■

- [1] M. Schuld, I. Sinayskiy, and F. Petruccione, The quest for a quantum neural network, *Quantum Inf. Process.* **13**, 2567 (2014).
- [2] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, [arXiv:1802.06002](https://arxiv.org/abs/1802.06002).
- [3] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, *Nature* **567**, 209 (2019).
- [4] I. Cong, S. Choi, and M. D. Lukin, Quantum convolutional neural networks, *Nat. Phys.* **15**, 1273 (2019).
- [5] M. Schuld and N. Killoran, Quantum machine learning in feature Hilbert spaces, *Phys. Rev. Lett.* **122**, 040504 (2019).
- [6] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, Power of data in quantum machine learning, *Nat. Commun.* **12**, 2631 (2021).
- [7] J. D. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature* **549**, 195 (2017).
- [8] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, *Quantum Sci. Technol.* **4**, 043001 (2019).
- [9] 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, *Nat. Rev. Phys.* **3**, 625 (2021).
- [10] J. Preskill, Quantum computing in the NISQ era and beyond, *Quantum* **2**, 79 (2018).
- [11] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke *et al.*, Noisy intermediate-scale quantum algorithms, *Rev. Mod. Phys.* **94**, 015004 (2022).
- [12] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, *Nat. Phys.* **17**, 1013 (2021).
- [13] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, The power of quantum neural networks, *Nat. Comput. Sci.* **1**, 403 (2021).
- [14] J. Jäger and R. V. Krems, Universal expressiveness of variational quantum classifiers and quantum kernels for support vector machines, *Nat. Commun.* **14**, 576 (2023).
- [15] Y. Du, Y. Yang, D. Tao, and M.-H. Hsieh, Problem-dependent power of quantum neural networks on multiclass classification, *Phys. Rev. Lett.* **131**, 140601 (2023).
- [16] Y. Wu, B. Wu, J. Wang, and X. Yuan, Quantum phase recognition via quantum kernel methods, *Quantum* **7**, 981 (2023).
- [17] E. Peters, J. Caldeira, A. Ho, S. Leichenauer, M. Mohseni, H. Neven, P. Spentzouris, D. Strain, and G. N. Perdue, Machine learning of high dimensional data on a noisy quantum processor, *npj Quantum Inf.* **7**, 161 (2021).
- [18] H.-L. Huang, Y. Du, M. Gong, Y. Zhao, Y. Wu, C. Wang, S. Li, F. Liang, J. Lin, Y. Xu *et al.*, Experimental quantum generative adversarial networks for image generation, *Phys. Rev. Appl.* **16**, 024051 (2021).
- [19] X. Pan, X. Cao, W. Wang, Z. Hua, W. Cai, X. Li, H. Wang, J. Hu, Y. Song, D.-L. Deng *et al.*, Experimental quantum end-to-end learning on a superconducting processor, *npj Quantum Inf.* **9**, 18 (2023).
- [20] W. Ren, W. Li, S. Xu, K. Wang, W. Jiang, F. Jin, X. Zhu, J. Chen, Z. Song, P. Zhang *et al.*, Experimental quantum adversarial learning with programmable superconducting qubits, *Nat. Comput. Sci.* **2**, 711 (2022).
- [21] J. Herrmann, S. M. Llima, A. Remm, P. Zapletal, N. A. McMahon, C. Scarato, F. Swiadek, C. K. Andersen, C. Hellings, S. Krinner *et al.*, Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases, *Nat. Commun.* **13**, 4144 (2022).
- [22] J.-y. Choi, S. Hild, J. Zeiher, P. Schauß, A. Rubio-Abadal, T. Yefsah, V. Khemani, D. A. Huse, I. Bloch, and C. Gross, Exploring the many-body localization transition in two dimensions, *Science* **352**, 1547 (2016).
- [23] M. Yan, H.-Y. Hui, M. Rigol, and V. W. Scarola, Equilibration dynamics of strongly interacting bosons in 2D lattices with disorder, *Phys. Rev. Lett.* **119**, 073002 (2017).
- [24] P. Bordia, H. Lüschen, S. Scherg, S. Gopalakrishnan, M. Knap, U. Schneider, and I. Bloch, Probing slow relaxation and many-body localization in two-dimensional quasiperiodic systems, *Phys. Rev. X* **7**, 041047 (2017).
- [25] M. Yan, H.-Y. Hui, and V. Scarola, Dynamics of disordered states in the Bose-Hubbard model with confinement, *Phys. Rev. A* **95**, 053624 (2017).
- [26] C. Balz, B. Lake, J. Reuther, H. Luetkens, R. Schönemann, T. Herrmannsdörfer, Y. Singh, A. Nazmul Islam, E. M. Wheeler, J. A. Rodriguez-Rivera *et al.*, Physical realization of a quantum spin liquid based on a complex frustration mechanism, *Nat. Phys.* **12**, 942 (2016).
- [27] M. Schreiber, S. S. Hodgman, P. Bordia, H. P. Lüschen, M. H. Fischer, R. Vosk, E. Altman, U. Schneider, and I. Bloch, Observation of many-body localization of interacting fermions in a quasirandom optical lattice, *Science* **349**, 842 (2015).
- [28] P. Bordia, H. P. Lüschen, S. S. Hodgman, M. Schreiber, I. Bloch, and U. Schneider, Coupling identical one-dimensional many-body localized systems, *Phys. Rev. Lett.* **116**, 140401 (2016).
- [29] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, *Phys. Rev. A* **99**, 032331 (2019).
- [30] I. J. Goodfellow, Y. Bengio, and A. C. Courville, Deep learning, *Nature* **521**, 436 (2015).
- [31] G. Verdon, M. Broughton, J. R. McClean, K. J. Sung, R. Babbush, Z. Jiang, H. Neven, and M. Mohseni, Learning to learn with quantum neural networks via classical neural networks, [arXiv:1907.05415](https://arxiv.org/abs/1907.05415).
- [32] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum natural gradient, *Quantum* **4**, 269 (2020).
- [33] J. Gacon, C. Zoufal, G. Carleo, and S. Woerner, Simultaneous perturbation stochastic approximation of the quantum Fisher information, *Quantum* **5**, 567 (2021).
- [34] B. van Straaten and B. Koczor, Measurement cost of metric-aware variational quantum algorithms, *PRX Quantum* **2**, 030324 (2021).
- [35] T. Volkoff and P. J. Coles, Large gradients via correlation in random parameterized quantum circuits, *Quantum Sci. Technol.* **6**, 025008 (2021).
- [36] R. Sweke, F. Wilde, J. Meyer, M. Schuld, P. K. Fährmann, B. Meynard-Piganeau, and J. Eisert, Stochastic

- gradient descent for hybrid quantum-classical optimization, *Quantum* **4**, 314 (2020).
- [37] Y. Du, M.-H. Hsieh, T. Liu, S. You, and D. Tao, Learnability of quantum neural networks, *PRX Quantum* **2**, 040337 (2021).
- [38] L. Bittel and M. Kliesch, Training variational quantum algorithms is NP-hard, *Phys. Rev. Lett.* **127**, 120502 (2021).
- [39] C. Wilson, J. Otterbach, N. Tezak, R. Smith, P. Karalekas, A. Polloreno, S. Alam, G. Crooks, and M. Da Silva, Quantum kitchen sinks: An algorithm for machine learning on near-term quantum computers, *Bull. Am. Phys. Soc.* **64**, 2 (2019).
- [40] T. Hur, L. Kim, and D. K. Park, Quantum convolutional neural network for classical data classification, *Quantum Mach. Intell.* **4**, 3 (2022).
- [41] J. Bowles, D. Wierichs, and C.-Y. Park, Backpropagation scaling in parameterised quantum circuits, [arXiv:2306.14962](https://arxiv.org/abs/2306.14962).
- [42] A. Abbas, R. King, H.-Y. Huang, W. J. Huggins, R. Movassagh, D. Gilboa, and J. McClean, On quantum backpropagation, information reuse, and cheating measurement collapse, in *Advances in Neural Information Processing Systems* **36**, (2024).
- [43] J. R. Glick, T. P. Gujarati, A. D. Corcoles, Y. Kim, A. Kandala, J. M. Gambetta, and K. Temme, Covariant quantum kernels for data with group structure, *Nat. Phys.* **20**, 479 (2024).
- [44] T. Hubregtsen, D. Wierichs, E. Gil-Fuster, P.-J. H. Derks, P. K. Faehrmann, and J. J. Meyer, Training quantum embedding kernels on near-term quantum computers, *Phys. Rev. A* **106**, 042431 (2022).
- [45] S. Thanasilp, S. Wang, M. Cerezo, and Z. Holmes, Exponential concentration in quantum kernel methods, *Nat. Commun.* **15**, 5200 (2024).
- [46] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag, Berlin, Heidelberg, 2007), 1st ed., <https://www.bibsonomy.org/bibtex/2d21de30a3a67c0f9f3c96bd6eec3267a/midtiby>.
- [47] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, *Quantum* **4**, 226 (2020).
- [48] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning, [arXiv:2001.03622](https://arxiv.org/abs/2001.03622).
- [49] M. Bilkis, M. Cerezo, G. Verdon, P. J. Coles, and L. Cincio, A semi-agnostic ansatz with variable structure for quantum machine learning, *Quantum Mach. Intell.* **5**, 43 (2023).
- [50] S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, Differentiable quantum architecture search, *Quantum Sci. Technol.* **7**, 045023 (2022).
- [51] Y. Du, T. Huang, S. You, M.-H. Hsieh, and D. Tao, Quantum circuit architecture search for variational quantum algorithms, *npj Quantum Inf.* **8**, 62 (2022).
- [52] F. Sauvage, M. Larocca, P. J. Coles, and M. Cerezo, Building spatial symmetries into parameterized quantum circuits for faster training, *Quantum Sci. Technol.* **9**, 015029 (2024).
- [53] M. Larocca, F. Sauvage, F. M. Sbahi, G. Verdon, and P. J. Coles, Group-invariant quantum machine learning, *PRX Quantum* **3**, 030341 (2022).
- [54] Y. Du, Y. Qian, X. Wu, and D. Tao, A distributed learning scheme for variational quantum algorithms, *IEEE Trans. Quantum Eng.* **3**, 1 (2022).
- [55] L. Banchi, J. Pereira, and S. Pirandola, Generalization in quantum machine learning: A quantum information standpoint, *PRX Quantum* **2**, 040321 (2021).
- [56] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, *Quantum* **5**, 582 (2021).
- [57] C. Gyurik, D. Vreumingen, and V. Dunjko, Structural risk minimization for quantum linear classifiers, *Quantum* **7**, 893 (2023).
- [58] Y. Du, Z. Tu, X. Yuan, and D. Tao, Efficient measure for the expressivity of variational quantum algorithms, *Phys. Rev. Lett.* **128**, 080506 (2022).
- [59] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles, Generalization in quantum machine learning from few training data, *Nat. Commun.* **13**, 4919 (2022).
- [60] X. Wang, Y. Du, Y. Luo, and D. Tao, Towards understanding the power of quantum kernels in the NISQ era, *Quantum* **5**, 531 (2021).
- [61] Y. Xue, X. Chen, T. Li, and S. H.-C. Jiang, in *International Conference on Machine Learning* (PMLR, 2023), p. 38881.
- [62] F. Qu, S. M. Erfani, and M. Usman, Performance analysis of coreset selection for quantum implementation of k-means clustering algorithm, [arXiv:2206.07852](https://arxiv.org/abs/2206.07852).
- [63] S. Otgonbaatar and M. Datcu, Assembly of a coreset of earth observation images on a small quantum computer, *Electronics* **10**, 2482 (2021).
- [64] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, Geometric approximation via coresets, *Comb. Comput. Geom.* **52**, 1 (2005).
- [65] O. Bachem, M. Lucic, and A. Krause, Practical coreset constructions for machine learning, [arXiv:1703.06476](https://arxiv.org/abs/1703.06476).
- [66] D. Feldman, Introduction to core-sets: An updated survey, [arXiv:2011.09384](https://arxiv.org/abs/2011.09384).
- [67] R. Z. Farahani and M. Hekmatfar, *Facility Location: Concepts, Models, Algorithms and Case Studies* (Springer-Verlag, Berlin, Heidelberg, 2009).
- [68] S. Jerbi, L. J. Fiderer, H. Poulsen Nautrup, J. M. Kübler, H. J. Briegel, and V. Dunjko, Quantum machine learning beyond kernel methods, *Nat. Commun.* **14**, 517 (2023).
- [69] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, *Phys. Rev. A* **103**, 032430 (2021).
- [70] M.-H. Yang, Face recognition using kernel methods, in *Advances in Neural Information Processing Systems* **14** (2001).
- [71] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, in *Advances in Neural Information Processing Systems* **31** (2018).
- [72] A. Barger and D. Feldman, in *Proceedings of the 2016 SIAM International Conference on Data Mining* (SIAM, Philadelphia, PA, 2016), p. 342.
- [73] H. Lu, Ph.D. thesis, The Pennsylvania State University, 2021.

- [74] G. Frahling and C. Sohler, in *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, USA, 2005), p. 209.
- [75] D. Feldman, A. Fiat, and M. Sharir, in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)* (IEEE, Berkeley, CA, 2006), p. 315.
- [76] H. Lu, M.-J. Li, T. He, S. Wang, V. Narayanan, and K. S. Chan, Robust coreset construction for distributed machine learning, *IEEE J. Sel. Areas Commun.* **38**, 2400 (2020).
- [77] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning* (MIT Press, Cambridge, MA, 2018).
- [78] H. Xiao, K. Rasul, and R. Vollgraf, Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, [arXiv:1708.07747](https://arxiv.org/abs/1708.07747).
- [79] D. Gross, S. T. Flammia, and J. Eisert, Most quantum states are too entangled to be useful as computational resources, *Phys. Rev. Lett.* **102**, 190501 (2009).
- [80] M. Yang, C.-l. Ren, Y.-c. Ma, Y. Xiao, X.-J. Ye, L.-L. Song, J.-S. Xu, M.-H. Yung, C.-F. Li, and G.-C. Guo, Experimental simultaneous learning of multiple nonclassical correlations, *Phys. Rev. Lett.* **123**, 190401 (2019).
- [81] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum-assisted quantum compiling, *Quantum* **3**, 140 (2019).