Quantum Information Processing (2024) 23:29
https://doi.org/10.1007/s11128-023-04231-7

Quantum kernel estimation-based quantum support vector regression

Xiaojian Zhou[1] · Jieyao Yu[1] · Junfan Tan[1] · Ting Jiang[2]

Received: 29 June 2023 / Accepted: 18 December 2023 / Published online: 23 January 2024

Abstract

Quantum machine learning endeavors to exploit quantum mechanical effects like superposition, entanglement, and interference to enhance the capabilities of classical machine learning methods. One of the most researched quantum machine learning methodologies presently is the quantum support vector machine (QSVM). Researchers have now developed quantum support vector classifiers and substantiated their potential for accelerating computation and enhancing classification accuracy in practical contexts through experimentation. Nevertheless, the utility of QSVM in regression tasks remains a relatively uncharted territory. In light of this, we introduce a quantum kernel estimation-based quantum support vector regression (QKE-QSVR) model for completing regression tasks. Within this proposed model, classical inputs are encoded as quantum feature vectors using the designed quantum feature map circuit with a variable parameter. The inner product between quantum feature vectors give rise to a trainable quantum kernel, which is subject to optimization through our proposed quantum kernel alignment-based regression (QKAR) algorithm, thereby bolstering the model's predictive accuracy when applied to a specific dataset. Subsequently, the trained quantum kernel is incorporated into the classical support vector regression process to construct the decision function and provide predictions for new data points. In this study, we validate the efficacy of the proposed model using three illustrative examples. Experimental findings underscore that, in comparison to the classical

✉ Ting Jiang
jiangtinghaha@126.com

Xiaojian Zhou
xjzhou@njupt.edu.cn

Jieyao Yu
yujieyao1029@163.com

Junfan Tan
TanJF0103@163.com

[1] Department of Management, Nanjing University of Posts and Telecommunications, Nanjing, China

[2] School of Information Engineering, Nanjing University of Finance and Economics, Nanjing, China

support vector machine model, our proposed model demonstrates superior predictive accuracy.

Keywords Quantum machine learning · Quantum kernel estimation · Quantum support vector regression · Quantum kernel alignment

1 Introduction

Quantum machine learning (QML) [1] has recently emerged as a prominent research orientation within the quantum information field. Its primary objective is to harness quantum mechanical effects, such as superposition, entanglement, and interference, to execute machine learning tasks. When compared to classical machine learning methodologies, QML presents potential advantages in several key dimensions. Firstly, QML offers a significant improvement in computational speed. A prototypical instance is the Shor's algorithm [2], devised to tackle the factorization problem of integers. Its optimal classical counterpart exhibits a computational complexity of $O\left(\exp\left(1.9(\log N)^{1/3}(\log\log N)^{2/3}\right)\right)$, which grows exponentially with the size of the target problem N. In contrast, the Shor's algorithm boasts a complexity of $O\left((\log N)^3\right)$, realizing a theoretical exponential speedup. Additional paradigmatic examples showcasing quantum acceleration advantages encompass the Grover search algorithm [3], the HHL algorithm [4], and the quantum support vector classification (QSVC) [5, 6]. Furthermore, QML exhibits a distinct advantage in terms of both generalization and expressive capabilities, as exemplified in generative models [7], quantum neural networks (QNN) [8–10], and QSVC [6, 11]. Finally, QML demands less training data or simpler models to comprehend complex relationships within the same dataset. Recent research suggests that an efficiently implementable QML model, one such that $T \in \mathcal{O}(\text{poly}n)$, only necessitates an effective amount of training data, $N \in \mathcal{O}(\text{poly}n)$, to attain robust generalization [12].

However, in practical terms, it is currently challenging to demonstrate the theoretical acceleration advantages of QML on noisy intermediate-scale quantum (NISQ) devices. This limitation arises from several factors inherent in NISQ devices. On the one hand, although the recent NISQ devices support over a hundred qubits, they still significantly fall short of meeting the quantum resource requirements for algorithms with theoretically exponential acceleration advantages, such as Shor's algorithm [2, 13, 14]. On the other hand, NISQ devices are susceptible to various errors, including decoherence, gate errors, measurement errors, crosstalk [15, 16], and only support a limited number of measurements. Nonetheless, a significant portion of theoretical outcomes [2, 6] is obtained within idealized settings, presupposing an infinite count of measurements and quantum systems devoid of noise—presumptions that prove impractical for contemporary NISQ devices in practice. Owing to these constraints of NISQ devices, the present stage of QML predominantly showcases its quantum advantage through heightened learning abilities in real-world applications [17–22]. In the forthcoming years, as the number of available quantum bits on NISQ devices rises

[23] and quantum error correction techniques advance [24, 25], there is an expectation that the theoretical advantages of quantum machine learning (QML) will be fully actualized.

One of the most prominently researched QML methods at present is the quantum support vector machine (QSVM). Lately, researchers have developed quantum support vector classifiers (QSVC) grounded in quantum computation. Unlike classical SVC, QSVC operates within an exponentially large quantum Hilbert space by transforming classical data into quantum states [26]. By leveraging the quantum Hilbert space as the feature space, QSVC can theoretically possess exponential speedup potential [27, 28]. For instance, [6] has demonstrated that QSVC can provide significant quantum acceleration for discrete logarithmic problems that are computationally challenging for classical approaches. However, practical implementations of QSVC face limitations due to the current constraints of NISQ computers, including finite qubit numbers and inherent device noise, which prevent the realization of its acceleration advantage in practical applications at present [15, 29]. Accordingly, present studies on QSVC mainly focus on verifying its advantages in classification accuracy, demonstrating that QSVC can produce better classification results than SVC since the quantum kernel has a better ability to capture the relationships between data [17, 18].

Presently, there exist two major pathways for constructing QSVC models: the variational quantum algorithm (VQA) and the quantum kernel estimation (QKE) [27]. VQA constructs the decision function directly within the quantum Hilbert space by utilizing a variational circuit. The circuit parameters are then optimized through training to minimize the designated cost function. Nevertheless, contemporary investigations indicate an exponential diminution in the gradient of VQAs as the number of qubits n [30–33], a phenomenon commonly recognized as barren plateaus in the training landscape. This phenomenon implies that an exponential number of measurements is requisite for precisely ascertaining the gradient to a predetermined precision, thereby imposing a significant resource overhead on the training regimen of VQAs.

In contrast, QKE focuses solely on estimating the inner product, known as the quantum kernel, between quantum states. This is achieved through techniques like the SWAP test [34] and inversion test [27]. Subsequently, the quantum kernel is employed in a classical SVC algorithm to determine the optimal Lagrange multipliers and generate predictions for new data points. Compared to VQA, QKE mitigates apprehensions associated with the task of pinpointing a fitting variational circuit ansatz and circumvents the necessity of tackling the predicament of traversing barren plateaus. Additionally, the convexity inherent in the training landscape guarantees that QKE can consistently ascertain the optimal model parameters throughout the training procedure [35]. The QKE method is currently being utilized to implement quantum support vector machines [27], quantum random forests [36], solve differential equations [37], predict and characterize second-order quantum phase transitions [38], and facilitate learning in Markov chains [39]. These studies suggest that QKE not only harbors theoretical potential for acceleration but also exhibits remarkable fitting and generalization capabilities in real-world applications. In consideration of the excellent characteristics of QKE, this work focusses on QKE.

To date, QKE-based QSVC (QKE-QSVC) has been successfully applied to high-energy physics [18, 40] and medical treatment [41], surpassing classical methods

with respect to classification accuracy [6, 28]. The application of QKE-QSVC in ligand-based virtual screening (LB-VS) exemplifies a standard case. LB-VS stands as a crucial methodology in the pursuit of novel therapeutic drugs. Nonetheless, LB-VS datasets are inherently complex, presenting a formidable challenge for conventional machine learning approaches to precisely discern potential drug candidates. Hence, [41] incorporates QKE-QSVC into the LB-VS drug discovery pipeline and showcases, employing ADRB2 and COVID-19 datasets [42], that the classification accuracy of QKE-QSVC surpasses that of classical state-of-the-art machine learning algorithms significantly.

Nonetheless, there has been limited research into the utilization of QKE methods for regression tasks. With this consideration, we extend QKE from classification to regression problems and propose a quantum kernel estimation-based quantum support vector regression (QKE-QSVR) model. In the proposed model, a trainable quantum kernel is employed, which is generated by introducing a variational parameter into the quantum feature map (QFM) circuit and trained by the proposed quantum kernel alignment regression (QKAR) method. The trained quantum kernel is then applied to a classical SVR algorithm to obtain the regression function. This investigation utilizes two simulated datasets generated by the Levy function N.13 [43] and the Ackley function [44], in addition to one real-life dataset—specifically, the NASA airfoil self-noise dataset [45]—to ascertain the feasibility and efficacy of the proposed QKE-QSVR model. Due to the difficulty of achieving the theoretical acceleration advantage on NISQ computers [16, 29, 46], similar to existing approaches, we do not compare the computational speed between the proposed QKE-QSVR and classical SVR; instead, our focus is on comparing the prediction accuracy between the two. The experimental results manifest that our QKE-QSVR model exhibits superior predictive accuracy on each of the three examples. This suggests that the predictive capabilities of QKE-QSVR can be systematically improved by constructing a quantum kernel. Furthermore, the approach presented in this study exhibits a certain level of generality, paving the way for potential applications in the enhancement of other classical machine learning models, like Kriging [47], in the future.

The remaining sections of this paper are structured as follows. In Sect. 2, we provide a concise overview of classical SVR and discuss its limitations. To address the challenges encountered by classical SVR, we introduce the QKE approach, which leverages the quantum kernel as its central component. Additionally, we introduce the quantum kernel alignment (QKA) method to enhance the expressiveness of the quantum kernel for a specific dataset. Moving on to Sect. 3, we illustrate the quantum circuit devised for this study and provide a brief outline of the proposed framework for the QKE-QSVR model. In Sect. 4, we verify the effectiveness of the proposed model through three examples conducted on Qiskit's Statevector Simulator. We compare the performance of our model with that of the classical SVR model and present the corresponding outcomes. Finally, in Sect. 5, we conclude this work with our final remarks, briefly outlining future research directions.

2 Existing theory

Prior to elucidating the QKE-QSVR model proposed in this paper, we undertake a comprehensive examination of the prevailing theories concerning SVR, QKE, and QKA, while establishing their interconnectedness. To commence, we provide an overview of SVR, focusing on its constraints in terms of kernel function computations. Subsequently, we introduce the QKE method as a means to tackle this challenge, enabling the computation of kernel functions within the quantum Hilbert space, specifically known as quantum kernels. Lastly, to enhance the representation capabilities of quantum kernels, we introduce the QKA method.

2.1 Support vector regression

SVR [48] is a highly recognized model in supervised learning, specifically designed for estimating regression in nonlinear systems. This study specifically focuses on the classical version of SVR, referred to as ε-SVR, which aims to identify a function $f(\mathbf{x})$ that minimizes the deviation of ε from the given target values y_i for the entire training dataset while maintaining a relatively flat function. More detailed insights into regression tasks and ε-SVR can be found in the works of [49, 50].

Suppose we are given a training sample set $T = \left\{\left(\mathbf{x}_i, y_i\right), i = 1, ..., l\right\}$, where $\mathbf{x}_i \in \mathbf{R}^N$ and $y_i \in \mathbf{R}$. Let's consider a regression function

$$f(\mathbf{x}) = \mathbf{w}^T \Phi(\mathbf{x}) + b \tag{1}$$

on a feature space $\mathcal{F}$, where $\mathbf{w}$ is a weight vector, b is the bias. SVM employs a nonlinear feature map $\Phi : \mathbf{R}^N \mapsto \mathcal{F}\Phi(\mathbf{x})$ to map a data point $\mathbf{x}$ into a feature vector $\Phi(\mathbf{x})$ in the feature space $\mathcal{F}$.

The function approximation problem can be expressed as the following optimization problem

$$\begin{aligned} &\min_{\mathbf{w},b} = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\left(\xi_i + \xi_i^*\right) \\ &\text{s.t}\begin{cases} y_i - \left(\mathbf{w}^T\Phi(\mathbf{x}) + b\right) \le \varepsilon + \xi_i \\ \mathbf{w}^T\Phi(\mathbf{x}) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases}, \end{aligned} \tag{2}$$

where $\varepsilon(> 0)$ represents the maximum permissible deviation, while $C(> 0)$ signifies the penalty associated with exceeding this deviation threshold. The slack variables, ξ_i and ξ_i^*, are employed to measure the extent of positive and negative deviations, respectively.

We can use the Lagrange multiplier method to obtain the dual form of the above optimization problem [51]

$$\min_{\alpha_i,\alpha_i^*} = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} R_{ij}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right) + \varepsilon\sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right) - \sum_{i=1}^{l} y_i\left(\alpha_i - \alpha_i^*\right)$$
$$\text{s.t.}\begin{cases} 0 \le \alpha_i, \alpha_i^* \le C, i = 1, ..., l \\ \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right) = 0 \end{cases}, \tag{3}$$

where α_i, α_i^* are Lagrange multiplier; $R_{ij} = \Phi(\mathbf{x}_i)^T\Phi\left(\mathbf{x}_j\right) = \kappa\left(\mathbf{x}_i, \mathbf{x}_j\right)$. Here $\kappa\left(\mathbf{x}_i, \mathbf{x}_j\right)$ represents the kernel function, which serves as a similarity metric that quantifies the level of resemblance between two feature vectors.

Then, the regression function is given by:

$$f(\mathbf{x}) = \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)\kappa(\mathbf{x}_i, \mathbf{x}) + b \tag{4}$$

By incorporating a kernel function, SVR can theoretically capture arbitrary nonlinear functions. However, there exist two major challenges. Firstly, the computational complexity of the kernel function generally scales with the number of training data points n, growing on the order of n^3 [52]. Secondly, choosing a suitable kernel function for a specific learning objective is a complex undertaking, as different kernel functions can result in different performance outcomes.

To address the above-mentioned challenges, researchers have proposed leveraging the power of quantum computing through the utilization of quantum circuits for generating and accessing useful kernel functions, a methodology known as the QKE approach. We will elaborate in the upcoming section. For more intricate details regarding QKE, please refer to [27].

2.2 Quantum kernel estimation

The QKE method is applied to classical data, which is initially encoded into a quantum state through a unitary operation (or a quantum circuit) denoted as $\mathcal{U}_\Phi(\mathbf{x})$. The operation $\mathcal{U}_\Phi(\mathbf{x})$ is contingent upon classical inputs $\mathbf{x} \in \mathcal{X}$. Once the data are incorporated, the quantum system assumes a state dependent on the pertinent data points. This progression is akin to the common definition of a feature map in classical machine learning. The operation $\mathcal{U}_\Phi(\mathbf{x})$ effectively maps a classical input $\mathbf{x}$ onto a quantum feature vector $|\Phi(\mathbf{x})\rangle$ within the quantum Hilbert space $\mathcal{F}$. Consequently, in this study, the transformation $\Phi : \mathcal{X} \to \mathcal{F}$ is denominated as the quantum feature map (QFM), and the corresponding quantum circuit $\mathcal{U}_\Phi(\mathbf{x})$ is denoted as the QFM circuit.

Definition 1 *(Quantum feature map)*—Consider a classical input set $\mathcal{X}$. Let $\mathcal{F}$ be the quantum Hilbert space. The quantum feature map is defined as the transformation from an input data $\mathbf{x} \in \mathcal{X}$ to a quantum feature vector $|\Phi(\mathbf{x})\rangle \in \mathcal{F}$.

$$\begin{aligned} \Phi &: \mathcal{X} \to \mathcal{F} \\ \Phi(\mathbf{x}) &= |\Phi(\mathbf{x})\rangle\langle\Phi(\mathbf{x})| = \rho(\mathbf{x}) \end{aligned} \tag{5}$$

which is implemented by a unitary operation $\mathcal{U}_\Phi(\mathbf{x})$:

$$|\Phi(\mathbf{x})\rangle = \mathcal{U}_\Phi(\mathbf{x})|0\rangle \tag{6}$$

Another alternative expression of QFM is:

$$\begin{aligned} &\Phi_v : \mathcal{X} \to \mathcal{F}_v \subset \mathcal{H} \otimes \mathcal{H}^* \\ &\Phi_v = |\Phi(\mathbf{x})\rangle \otimes |\Phi^*(\mathbf{x})\rangle \end{aligned}, \tag{7}$$

where $|\Phi^*(\mathbf{x})\rangle$ represents the quantum state generated through the application of the complex conjugate unitary operator $|\Phi^*(\mathbf{x})\rangle = \mathcal{U}_\Phi^*(\mathbf{x})|0\rangle$. Meanwhile, $\mathcal{F}_v$ denotes the tensor product space comprising a data-encoding Dirac vector and its complex conjugate.

The inner product between quantum feature vectors $|\Phi(\mathbf{x}_i)\rangle$, $\left|\Phi\left(\mathbf{x}_j\right)\right\rangle$ gives rise to a quantum kernel:

Definition 2 *(Quantum kernel)*—Given a quantum feature map $\Phi : \mathcal{X} \to \mathcal{F}$. Its corresponding quantum kernel is fundamentally the inner product between quantum feature vectors $|\Phi(\mathbf{x}_i)\rangle$, $\left|\Phi\left(\mathbf{x}_j\right)\right\rangle$ with $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$:

$$\kappa\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left|\langle\Phi(\mathbf{x}_i)|\,\Phi\left(\mathbf{x}_j\right)\right\rangle\right|^2 \tag{8}$$

Owing to the existence of device noise in NISQ computers, the quantum states obtained through a QFM cannot be assured to be in pure states. Therefore, it becomes imperative to contemplate an alternative representation of quantum states, specifically, the density matrix $\rho(\mathbf{x}) = |\Phi(\mathbf{x})\rangle\langle\Phi(\mathbf{x})|$.

Quantum kernel, as represented by the density matrix, is delineated as follows:

$$\kappa\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left|\langle\Phi(\mathbf{x}_i)|\,\Phi\left(\mathbf{x}_j\right)\right\rangle\right|^2 = tr\left[\rho(\mathbf{x}_i)\rho\left(\mathbf{x}_j\right)\right] \tag{9}$$

Example For clarity, we present a straightforward example illustrating the concept of a quantum kernel. Assuming the application of Pauli X rotation gate $R_X(\mathbf{x}) = e^{-i\frac{\mathbf{x}}{2}\sigma_\mathbf{x}}$ to map a classical data $\mathbf{x}$ into the following quantum state.

$$|\psi(\mathbf{x})\rangle = R_X(\mathbf{x})|0\rangle = \cos\left(\frac{\mathbf{x}}{2}\right)|0\rangle + i\,\sin\left(\frac{\mathbf{x}}{2}\right)|1\rangle \tag{10}$$

The density matrix representation of the aforementioned quantum state is given by:

$$\begin{aligned} \rho_X(\mathbf{x}) = &\cos^2\left(\frac{\mathbf{x}}{2}\right)|0\rangle\langle 0| + i\cos\left(\frac{\mathbf{x}}{2}\right)\sin\left(\frac{\mathbf{x}}{2}\right)|0\rangle\langle 1| - i\cos\left(\frac{\mathbf{x}}{2}\right)\sin\left(\frac{\mathbf{x}}{2}\right)|1\rangle\langle 0| \\ &+ \sin^2\left(\frac{\mathbf{x}}{2}\right)|1\rangle\langle 1|, \end{aligned} \tag{11}$$

and the quantum kernel can then be derived:

$$K_{R_X}(\mathbf{x}_i, \mathbf{x}_j) = \left|\cos\left(\frac{\mathbf{x}_i}{2}\right)\cos\left(\frac{\mathbf{x}_j}{2}\right) + \sin\left(\frac{\mathbf{x}_i}{2}\right)\sin\left(\frac{\mathbf{x}_j}{2}\right)\right|^2 = \cos\left(\frac{\mathbf{x}_i - \mathbf{x}_j}{2}\right)^2 \quad (12)$$

It is evident that the application of the Pauli X rotation gate for mapping classical data into the quantum feature space yields a translation-invariant squared cosine kernel.

Subsequently, we commence the discussion on the quantum kernel computation using a quantum computer. One straightforward approach is the inversion test [27]. In this case, we utilize the fact that the kernel entries of samples $\mathbf{x}_i, \mathbf{x}_j$ can be expressed as the transition amplitude:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \left|\langle 0^{\otimes n}|\mathcal{U}^\dagger_{\Phi(\mathbf{x}_j)}\mathcal{U}_{\Phi(\mathbf{x}_i)}|0^{\otimes n}\rangle\right|^2 \quad (13)$$

To obtain an estimate, the quantum circuit $\mathcal{U}^\dagger_{\Phi(\mathbf{x}_j)}\mathcal{U}_{\Phi(\mathbf{x}_i)}$ is first applied to the initial state $|0^{\otimes n}\rangle$. Subsequently, R samplings are performed on the resulting state $\mathcal{U}^\dagger_{\Phi(\mathbf{x}_j)}\mathcal{U}_{\Phi(\mathbf{x}_i)}|0^{\otimes n}\rangle$ in the computational basis. The occurrences of all observed zero R bit-strings (0, ..., 0) are recorded and then divided by the total number of samplings. The resulting frequency then serves as an estimator for the quantum kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j)$, subject to the influence of sampling errors $\tilde{\varepsilon} = \mathcal{O}(R^{-1/2})$.

It is crucial to emphasize that the performance of the model is significantly influenced by the QFM. Recent studies suggest that the use of unstructured QFMs might jeopardize the overall generalization and trainability of the QML model [53]. Consequently, it is prudent to abstain from employing QFM that lacks relevance to the particular problem, and thoughtful attention should be dedicated to the problem's data structure during QFM design [54–56]. Nevertheless, constructing problem-motivated QFM can be challenging in many cases. An alternative approach involves employing a parameterized QFM $\Phi(\mathbf{x}, \lambda)$ with a trainable rotation parameter λ, which gives rise to a parameterized quantum kernel $\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j)$ [19, 26]. Through the training of the parameterized quantum kernel $\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j)$, one can effectively enhance the model's generalization performance for a specific dataset [19, 26].

2.3 Quantum kernel alignment

In this section, we present a detailed exposition on quantum kernel training. Presently, a straightforward technique for kernel optimization involves exhaustive search. Nevertheless, the computational complexity of this approach scales exponentially $\mathcal{O}(\exp(r))$, concerning the dimensionality r of the parameter vector, making it practical only for a limited number of parameters. Recognizing this, [57] employs classical kernel alignment theory [58, 59] to refine the trainable parameter based on training data, thereby acquiring a quantum kernel with ideal generalization capability. This approach is referred to as quantum kernel alignment (QKA).

Before delving into a detailed exposition of QKA, let us provide a concise introduction to the theory of kernel alignment. Kernel alignment is a widely employed metric

for assessing the quality of a kernel function, aiming to minimize the generalization error bound of SVM. Shawe-Taylor and Cristianini [60] established the upper bound for the generalization error of SVM based on the fat-shattering dimension of linear threshold functions, which can be derived from the original objective function of SVM [48]:

$$P(y \neq f(z)) \leq \tilde{\mathcal{O}}\Big(m^{-1} F^*(\lambda)\Big), \tag{14}$$

where λ denotes the variational parameter, $F(\lambda)$ is the original objective function of the SVM.

Minimizing the upper bound of the generalization error in Eq. (14) is equivalent to solving the following optimization problem

$$\min_{\lambda} \max_{\alpha} F(\alpha, \lambda), \tag{15}$$

where α is the solution to the convex optimization problem of SVM, λ denotes the variational parameter involved in the quantum circuit, $F(\alpha, \lambda)$ is the cost function given by the objective function of SVM. In other word, the kernel alignment can be viewed as the pursuit of an optimal kernel that results in the lowest possible generalization error for the learning model. For detailed theoretical derivations regarding kernel alignment, refer to [58, 59].

To tackle the optimization problem described in Eq. (15), QKA constructs an iterative algorithm based on the hybrid quantum–classical (HQC) workflow. In each iteration, the quantum computation component is employed for the calculation of the quantum kernel matrix $\kappa_\lambda\left(\mathbf{x}_i, \mathbf{x}_j\right)$. Subsequently, the classical SVM procedure invokes $\kappa_\lambda\left(\mathbf{x}_i, \mathbf{x}_j\right)$ to solve for the Lagrange multipliers α. Following the acquisition of $\kappa_\lambda\left(\mathbf{x}_i, \mathbf{x}_j\right)$ and α, Eq. (15) is utilized as the loss function. Leveraging a classical simultaneous perturbation stochastic approximation (SPSA) [61] optimizer, the gradient of the loss function with respect to α is evaluated, and the variational parameter λ is updated accordingly. The ultimately optimized α^* and λ^* can then be utilized within the classical SVM procedure for predicting test data.

The choice of SPSA as the optimizer is substantiated for the following reasons. Firstly, it necessitates significantly fewer circuit executions to achieve results comparable to gradient-based methods, presenting the potential for time and resource savings. Secondly, it exhibits robustness against the impacts of device noise in NISQ quantum computers [62].

3 The proposed QKE-QSVR model

In this section, we begin by describing the quantum circuit used in this paper. Following that, we provide a brief overview of the overall framework of the proposed QKE-QSVR model.

3.1 Design of quantum circuit

The QFM circuit $\mathcal{U}_\Phi(\mathbf{x})$ is a crucial component of the QKE-QSVR model, as it determines the model's capacity to capture the underlying structure of the data. It can be expressed in the following form:

$$\mathcal{U}_\Phi(\mathbf{x}) = \prod_d U_{\Phi(\mathbf{x})} H^{\otimes n}, \tag{16}$$

where d denotes the circuit depth, H is the Hadamard gate, $U_{\Phi(\mathbf{x})}$ represents the Pauli expansion matrix

$$U_{\Phi(\mathbf{x})} = \exp\left(i \sum_{S \subseteq [n]} \Phi_S(\mathbf{x}) \prod_{i \in S} P_i \right) \tag{17}$$

with a nonlinear encoding function

$$\Phi_S(\mathbf{x}) = \begin{cases} \mathbf{x}_j & \text{if } S = \{\mathbf{x}_j\} \\ \prod_{j \in S} (\pi - \mathbf{x}_j) & \text{if } |S| > 1 \end{cases} \tag{18}$$

here S describes the associations between different qubits. $P_i \in \{I, X, Y, Z\}$ represents the unitary Pauli rotation transformation for X, Y, Z, which captures Ising-like interactions (i.e., ZZ) as well as non-interacting terms (i.e., X, Z, Y).

For a given learning objective, diverse QFMs may yield distinct data distributions in the quantum feature space, thereby influencing the performance of the model. To identify a suitable QFM for the QKE-QSVR model, we explore four widely used Pauli expansion QFMs: Z feature map, ZZ feature map, X feature map, and Y feature map, as illustrated in Table 1. These four QFMs were implemented using Qiskit's circuit library [63].

Here, we specifically expound upon the ZZ feature map with Pauli sequence $[Z, ZZ]$, and the corresponding quantum circuit is illustrated in Fig. 1. The Pauli expansion matrix for the ZZ feature map can be expressed as:

$$\mathcal{U}_\Phi(\mathbf{x}) = \exp(i\mathbf{x}_0 Z_0 + i\mathbf{x}_1 Z_1 + i(\pi - \mathbf{x}_0)(\pi - \mathbf{x}_1) Z_0 Z_1) \tag{19}$$

Table 1 The Pauli expansion of four quantum feature maps: Z, ZZ, X, Y

Quantum feature map	Pauli matrices
Z Feature map	Z
ZZ Feature map	Z, ZZ
X Feature map	X
Y Feature map	Y

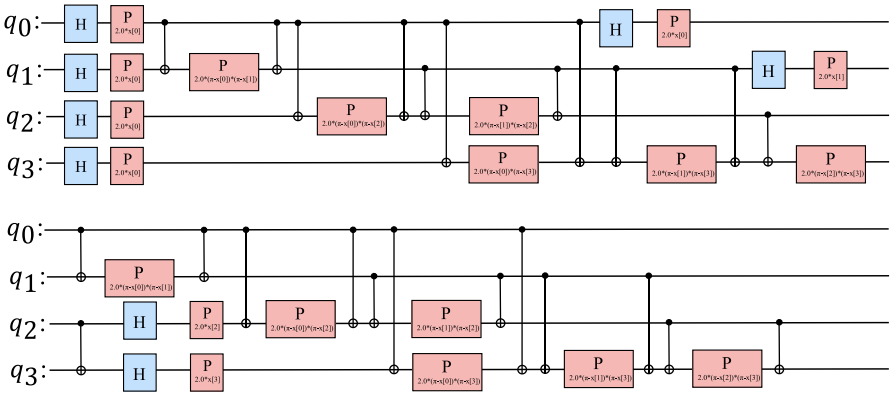

Fig. 1 Quantum circuit of the ZZ feature map with four qubits, which consists of single-qubit rotation gates and two-qubit CNOT entangling gates

Specifically, $\exp(i\mathbf{x}_0 Z_0) = R_Z(2\mathbf{x}_0)$, $\exp(i\mathbf{x}_1 Z_1) = R_Z(2\mathbf{x}_1)$, and the tensor product $\exp(i(\pi - \mathbf{x}_0)(\pi - \mathbf{x}_1)Z_0 Z_1) = CX(I \otimes R_Z(2(\pi - \mathbf{x}_0)(\pi - \mathbf{x}_1)))CX$.

After determining the suitable QFM, a variational parameter λ is introduced in the QFM circuit to generate a trainable quantum kernel. The overall quantum circuit employed in this study is illustrated in Fig. 2, and the resulting quantum kernel is given by:

$$\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j) = \left|\langle 0^{\otimes n}| R_Y^\dagger(\lambda)\mathcal{U}_{\Phi(\mathbf{x}_j)}^\dagger \mathcal{U}_{\Phi(\mathbf{x}_i)} R_Y(\lambda)|0^{\otimes n}\rangle\right|^2, \tag{20}$$

where $R_Y(\lambda) = \exp(-i(\lambda/2)Y)$ for a parameter $\lambda \in [0, 2\pi]$.

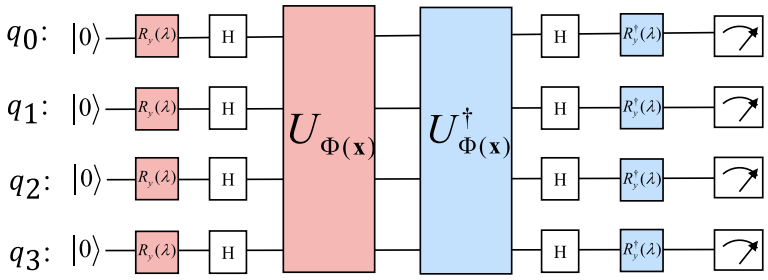

Fig. 2 Overall quantum circuit employed in this study. H is a Hadamard gate, $U_{\Phi(\mathbf{x})}$ is a unitary operator, $R_y(\lambda)$ is a embedding unitaries

3.2 Framework of QKE-QSVR model

Our QKE-QSVR model follows the HQC workflow, with the quantum computation component being utilized solely for computing the quantum kernel, while other common tasks such as data preprocessing and parameter optimization are carried out on a classical computer.

Specially, during the training phase, preprocessed training data undergo encoding into quantum states through the QFM. The quantum computation component is employed to evaluate the quantum kernel matrix $\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j)$ for all pairs of training data. Following this, the variational parameter λ undergoes optimization via QKA, yielding the trained quantum kernel matrix $\kappa_{\lambda^*}(\mathbf{x}_i, \mathbf{x}_j)$ with optimal λ^*. The detailed steps of training the quantum kernel matrix will be expounded upon in subsequent sections. Lastly, in the classical component, the trained quantum kernel matrix $\kappa_{\lambda^*}(\mathbf{x}_i, \mathbf{x}_j)$ is utilized in a classical SVR procedure to solve the convex quadratic programming problem in Eq. (3), yielding the parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, b and the support vectors $\{\hat{\mathbf{x}}_j\}_{j=1}^m$. The hyperparameters C, ε of SVR are tuned using classical cross-validation.

During the testing phase, both test and training data are encoded into quantum states via the QFM. The quantum kernel $\kappa_{\lambda^*}(\mathbf{x}'_k, \hat{\mathbf{x}}_i)$ between a test data $\mathbf{x}'_k \in S$ and all support vectors $\{\hat{\mathbf{x}}_j\}_{j=1}^m$ is evaluated utilizing the quantum computation component. Subsequently, based on the optimal decision function obtained during the training phase, predictions can be made for the test data.

The framework of QKE-QSVR model is as follows

Step 1 Perform the data preprocessing step. The dataset is partitioned into a training set $T = \{(\mathbf{x}_i, y_i)\}$, $i = 1, ..., l$ and a testing set $S = \{\mathbf{x}'_k\}$, $k = 1, ..., s$.

Step 2 Select a suitable QFM for the given dataset. We compare the performance of four commonly used QFMs as presented in Sec 3.1 and select the one with best predictive accuracy for subsequent steps.

Step 3 Train the quantum kernel. A variational parameter λ is introduced in the QFM circuit, which gives rise to a trainable quantum kernel

$$\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j) = \left|\langle 0^{\otimes n}| R_Y^\dagger(\lambda)\mathcal{U}_{\Phi(\mathbf{x}_j)}^\dagger \mathcal{U}_{\Phi(\mathbf{x}_i)} R_Y(\lambda)|0^{\otimes n}\rangle\right|^2 \tag{21}$$

Next, we perform the proposed *quantum kernel alignment-based regression* (QKAR) algorithm presented below to train the variational parameter λ and thus obtain the optimal quantum kernel.

Step 4 Train the QKE-QSVR model. Apply the optimal quantum kernel $\kappa_{\lambda^*}(\mathbf{x}_i, \mathbf{x}_j)$ obtained in Step 3 to a classical SVR algorithm for getting the optimal solution $\boldsymbol{\alpha} = \left(\alpha_1^{(*)}, \alpha_2^{(*)}, ..., \alpha_n^{(*)}\right)$ of the convex optimization problem given in Eq. (3).

Step 5 Predict the test data. Calculate the quantum kernel $\kappa_{\lambda^*}(\mathbf{x}'_k, \hat{\mathbf{x}}_i)$ between each test data $\mathbf{x}'_k \in S$ and all support vectors $\{\hat{\mathbf{x}}_i\}_{i=1}^m$ with the optimal parameter λ^* to obtain the prediction for the test data:

$$f(\mathbf{x}_k\prime) = \sum_{i=1}^{m} (\hat{\alpha}_i - \hat{\alpha}_i^*)\kappa_{\lambda^*}(\mathbf{x}_k\prime, \hat{\mathbf{x}}_i) + b \tag{22}$$

The algorithm of QKAR is as follows

Step 1 Initialize the variational parameter $\lambda = \lambda_0$ and determine the maximum number of iterations j.

Step 2 Randomly generate a perturbation vector $\Delta_k \in \{-1, 1\}$.

Step 3 Create two new version of the rotation parameter vector $\lambda_{+,k} = \lambda_k + c_k\Delta_k$, $\boldsymbol{\lambda}_{-,k} = \boldsymbol{\lambda}_k - c_k\Delta_k$ from the perturbation vector, where $c_k = \frac{c}{(k+1)^\gamma}$ for constants c, γ.

Step 4 Evaluate two quantum kernels $\kappa_+ = \kappa_{\lambda_{+,k}}(\mathbf{x}_i, \mathbf{x}_j)$ and $\kappa_- = \kappa_{\lambda_{-,k}}(\mathbf{x}_i, \mathbf{x}_j)$ with all training data utilizing the QKE subroutine.

Step 5 Obtain the solution $\alpha_{\pm,k}$ of the convex optimization problem in Eq. (3) using the classical optimizer.

Step 6 Perform two evaluations of the loss function to estimate its gradient with respect to λ_k:

$$g_k(\alpha_k, \lambda_k) = \frac{1}{2c_k\Delta_{k_i}}\left[F\left(\alpha_{+,k}, \lambda_{+,k}\right) - F\left(\alpha_{-,k}, \lambda_{-,k}\right)\right] \tag{23}$$

The loss function is defined as according to the kernel alignment theory:

$$F(\lambda) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\kappa_\lambda(\mathbf{x}_i, \mathbf{x}_j)(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right) + \varepsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \tag{24}$$

Step 7 Determine whether the cost function has converged or whether the maximum number of iterations has been reached. If either of these two conditions is satisfied, proceed to Step 9; otherwise, proceed to Step 8.

Step 8 Update λ depending on the gradient $g_k(\alpha_k, \lambda_k)$ and learning rate a_k and then go to Step 2. The update rule is:

$$\lambda_{k+1} = \lambda_k - a_k g_k(\alpha_k, \lambda_k) \tag{25}$$

Step 9 Terminate the algorithm and output the optimal quantum kernel $\kappa_{\lambda^*}(\mathbf{x}_i, \mathbf{x}_j)$.

The details of the proposed QKE-QSVR model are presented above. Our QKE-QSVR model solves the regression problem by employing QFM with a variational parameter to exploit the trainable quantum kernel. This variational parameter is then trained with a given training set utilizing the proposed QKAR algorithm. The QKAR algorithm employs kernel alignment as its loss function. Minimizing this loss function is akin to minimizing the generalization error bound of SVR [58, 59]. Consequently, training the quantum kernel using QKAR can theoretically enhance the model's predictive capability for unknown data. In addition, the QKAR algorithm employs SPSA to optimize the trainable parameter. The computational complexity of SPSA is independent of the number of free parameters. Therefore, compared to commonly used kernel optimization methods such as exhaustive search (which experiences exponential growth in computational complexity $\mathcal{O}(\exp(r))$ with the dimensionality of the parameter vector r), QKAR is better suited for quantum kernel training. To validate the

above-mentioned advantages, experiments are conducted on three different datasets. The detailed procedure and results are presented in the next section.

4 Examples

In this section, we present a 2-qubit example, an 8-qubit example and a real-life example on the Statevector Simulator from the IBM Quantum framework [63] to validate the effectiveness of the proposed QKE-QSVR model. The Qiskit's Statevector Simulator is widely used in the field of QML as it helps to reduce the consumption of quantum resources while providing reliable and stable results.

In addition, it is worth noting that due to the limited number of qubits supported by NISQ computers and the unavoidable device noise, the current implementation of QKE-QSVC cannot achieve theoretical quantum acceleration [15, 16, 46]. Therefore, the experimental researches on QKE-QSVC in the current stage focus only on comparing its prediction accuracy with classical SVC [17–19]. Consequently, similar to existing studies, we do not compare the training speed between the proposed QKE-QSVR model and classical SVR, but rather compares their prediction performance.

4.1 A 2-qubit example

In this example, we utilize the Levy function N.13 [43] to confirm the effectiveness of the QKE-QSVR model. The Levy function N.13 is a widely used function to assess the proficiency of machine learning models, which is defined on $x_i \in [-10, 10], i = 1, 2$. Its expression is as follows:

$$f(x) = \sin^2(3\pi x_1) + (x_1 - 1)^2\Big[1 + \sin^2(3\pi x_2)\Big] + (x_2 - 1)^2\Big[1 + \sin^2(2\pi x_2)\Big] \tag{26}$$

To validate the effectiveness of the proposed model under various training sample sizes, we carry out two sets of experiments. Both experiments utilize datasets generated by the Levy function N.13, with varying numbers of training samples. In the first experiment, we use a training set of 20 samples and a test set of 5 samples. For the second experiment, we employ a training set of 80 samples and a test set of 20 samples. Below, we will elaborate on the experimental procedures in detail.

Our QKE-QSVR model is a HQC algorithm designed to process classical data. In order to achieve this, classical data must first be encoded into quantum states via a QFM circuit. Selecting an applicable QFM is of great significance for a particular dataset. This is because different QFMs can capture specific types of information, such as nonlinear relationships, potentially resulting in varying model performance [64–67]. Unfortunately, knowing in advance which QFM will yield the best performance for a given dataset can be a challenging task. Hence, we compare the performance of four commonly used QFMs (Z feature map, ZZ feature map, X feature map, Y feature map). We then select the QFM that yields the best performance for the subsequent procedures. For this study, we utilize root-mean-square error (RMSE) and mean absolute error

(MAE) as metrics to measure the model's performance. RMSE stands for the square root of the average squared differences between predictive and actual, while MAE represents the average absolute error between predictive and actual values. These two metrics can efficiently evaluate the predictive ability of different models. The expressions for RMSE and MAE are as follows:

$$\mathrm{RMSE} = \sqrt{\sum_{i=1}^{n} (y - f(x))^2 / n} \tag{27}$$

$$\mathrm{MAE} = \sum_{i=1}^{n} |y - f(x)| / n \tag{28}$$

Upon identifying the QFM with the optimal performance, its corresponding quantum kernel is utilized in a classical SVR algorithm. It should be noted that the quantum kernel generated by the selected QFM but not yet subjected to training is denoted as the untrained quantum kernel in this paper.

Afterward, to further enhance the performance of the model for the given dataset, a variable parameter λ is introduced in the QFM circuit to obtain a trainable quantum kernel, which is then optimized using the QKAR algorithm. In this example, the ideal initial value of the variational parameter λ is set to $\pi/2$, and the maximum number of iterations is limited to 30. The detailed process of getting the optimal quantum kernel using the QKAR algorithm is described in Sect. 3.2. For ease of distinction, the optimal quantum kernel obtained through the QKAR algorithm is referred to as the trained quantum kernel, which is then applied in a classical SVR algorithm to find the optimal Lagrange multipliers. In this example, the hyperparameters of the QKE-QSVR model are adjusted through cross-validation. Table 2 displays the specific hyperparameter values for the QKE-QSVR model trained with 20 and 80 samples, respectively.

Next, to validate the superiority of the proposed QKE-QSVR model, we compare its performance with that of the classical SVR model based on three widely used traditional kernel functions (linear, poly, RBF). To ensure fairness, we adjust the hyperparameters C, ε, σ of the classical SVR model utilizing cross-validation. Table 3 displays the precise hyperparameter values for the classical SVR model trained with 20 and 80 samples, respectively.

In order to alleviate the impact of stochasticity, the aforementioned processes are repeated ten times in this example, with the average results being deemed as the ultimate outcomes, which are presented in Tables 4 and 5.

Table 2 Hyperparameter values of the QKE-QSVR model trained with 20 and 80 samples in the 2-qubit example

Training samples (n)	Hyperparameter	
	C	ε
20	0.1	10.0
80	0.01	10.0

Table 3 Hyperparameter values of the classical SVR model trained with 20 and 80 samples in the 2-qubit example

Training samples (n)	Hyperparameter		
	C	ε	σ
20	0.1	10.0	5.0
80	0.01	10.0	3.5

Table 4 Average RMSE and MAE achieved by the quantum and classical kernels when trained with 20 samples in the 2-qubit example

Type	Kernel	RMSE	MAE
Quantum kernel	Untrained quantum kernel	0.2815	0.2299
	Trained quantum kernel	**0.2224**	**0.1889**
Classical kernel	Poly	0.5323	0.4264
	Linear	0.6190	0.5025
	RBF	**0.3104**	**0.2440**

The bold indicates the lowest RMSE and MAE values

Table 5 Average RMSE and MAE achieved by the quantum and classical kernels when trained with 80 samples in the 2-qubit example

Type	Kernel	RMSE	MAE
Quantum kernel	Untrained quantum kernel	0.1836	0.1359
	Trained quantum kernel	**0.1602**	**0.1250**
Classical kernel	Poly	0.4902	0.3889
	Linear	0.5595	0.4157
	RBF	**0.1996**	**0.1425**

The bold indicates the lowest RMSE and MAE values

We first discuss the results obtained from experiments utilizing 20 training samples, which are presented in Table 4. It can be observed that the trained quantum kernel yields a decrease of 20.99% in the average value of RMSE and a decrease of 17.83% in the average value of MAE compared to the untrained one. The results imply that training the quantum kernel with the QKAR algorithm can effectively enhance the predictive accuracy of the model.

We also compare the performance of the proposed QKE-QSVR model using the trained quantum kernel with that of the classical SVR model based on three traditional kernel functions. The average values of RMSE and MAE in our QKE-QSVR model (with the trained quantum kernel) are 0.2224 and 0.1889, respectively. In contrast, the minimum average values of RMSE and MAE of the classical SVR model (with the RBF kernel) are 0.3104 and 0.2440, respectively. These findings indicate that the proposed QKE-QSVR model significantly outperforms the classical SVR model in

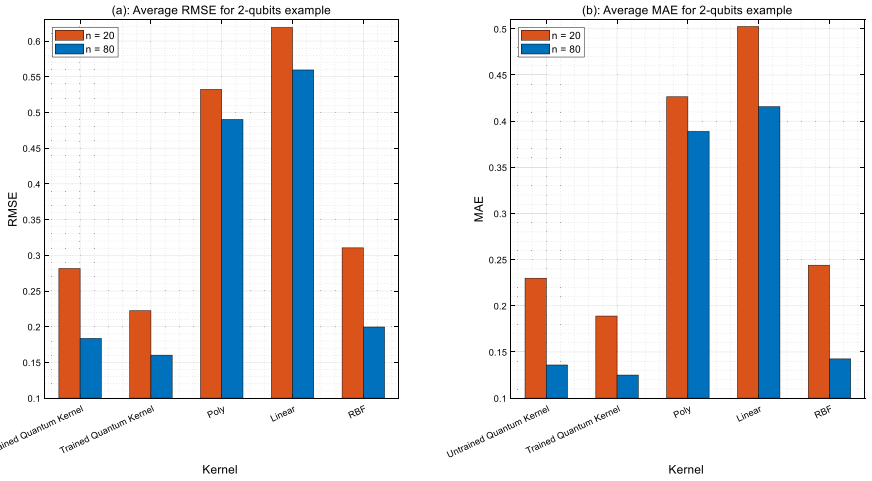

Fig. 3 Average RMSE (**a**) and MAE (**b**) achieved by quantum and classical kernels using 20 (red) and 80 (blue) training samples in the 2-qubit example (Color figure online)

terms of predictive accuracy, highlighting the superior ability of the trained quantum kernel to express the relationships between data.

Next, we will discuss the results obtained from experiments utilizing 80 training samples, which are presented in Table 5. It is worth noting that the trained quantum kernel has significantly reduced the average value of RMSE value by 12.75% and the average value of MAE by 8.02%. This improvement indicates that the trained quantum kernel provides better expressiveness compared to the untrained one. On the other hand, in contrast to the classical SVR model, the proposed QKE-QSVR model (with the trained quantum kernel) achieves an average RMSE and MAE of 0.1602 and 0.1250, respectively. Meanwhile, the smallest average values of RMSE and MAE achieved by the classical SVR model (with the RBF kernel) are 0.1996 and 0.1425, respectively. These experimental results confirm the superior predictive accuracy of the proposed QKE-QSVR model over the classical SVR model.

Compared to the results obtained from models trained with 20 and 80 samples (Fig. 3), we observe that the prediction accuracy of the QKE-QSVR model is higher than that of the classical SVR model, regardless of whether 20 or 80 training samples are used. Moreover, as depicted in Fig. 3, it is evident that the QKE-QSVR model trained with 80 training samples exhibits superior predictive capabilities. This observation suggests that with an increasement in the volume of data, the quantum kernel more comprehensively captures relationship within the data, consequently elevating the model's accuracy to a more representative level.

4.2 An 8-qubit example

In this case, the Ackley function [44] is utilized to test the performance of the propose QKE-QSVR model, which is usually evaluated on the hypercube $x_i \in$

Table 6 Hyperparameter values of the QKE-SVR model trained with 20 and 80 samples in the 8-qubit example

Training samples (n)	Hyperparameter	
	C	ε
20	10.0	0.1
80	0.1	10.0

$[-32.768, 32.768]$ for all $i = 1, ..., 8$. The expression of the Ackley function is as follows:

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{8}\sum_{i=1}^{8}x_i^2}\right) - \exp\left(\frac{1}{8}\sum_{i=1}^{8}\cos(2\pi x_i)\right) + 20 + \exp(1) \tag{29}$$

Similar to the 2-qubit example, two sets of experiments are performed to verify the efficacy of the proposed model across different training sample sizes. The datasets utilized in both experiments are generated through the Ackley function, with varying numbers of training samples. In the first experiment, a training set of 20 samples and a test set of 5 samples are used. In the second experiment, a larger training set consisting of 80 samples and a test set of 20 samples are employed.

We commence by comparing the performance of the aforementioned QFMs to determine which one is best-suited for the given dataset. The QFM that yields the highest predictive accuracy is selected for subsequent operations. Its corresponding quantum kernel, namely the untrained quantum kernel, is then utilized in a classical SVR algorithm. Next, to further improve the model's performance on the given dataset, we introduce a variable parameter in the QFM circuit and train it using the QKAR algorithm. Finally, the trained quantum kernel is applied in a classical SVR algorithm to find the optimal regression function. Similar to the 2-qubit example, the hyperparameters of the QKE-QSVR model are tuned through cross-validation. The specific hyperparameter values for the QKE-QSVR model trained with 20 and 80 samples are shown in Table 6.

In parallel, we demonstrate the superiority of the proposed QKE-QSVR model by comparing its performance with that of the classical SVR model based on three traditional kernel functions. For sake of fairness, we adjust the hyperparameters C,ε, σ of the classical SVR model utilizing cross-validation. The specific hyperparameter values for the QKE-QSVR model trained with 20 and 80 samples are shown in Table 7.

Similar to the 2-qubit example, we repeat the above procedures ten times in the 8-qubit example and take the average results as the final outcomes, which are shown in Tables 8 and 9.

Based on the results obtained from experiments utilizing 20 training samples, as shown in Table 8, we observe that the trained quantum kernel reduces the average RMSE by 19.25% and the average MAE by 21.61% compared to the untrained quantum kernel. This demonstrates that training the quantum kernel using the QKAR algorithm can effectively improve the model's predictive accuracy. Additionally, we

Table 7 Hyperparameter values of the classical SVR model trained with 20 and 80 samples in the 8-qubit example

Training samples (n)	Hyperparameter		
	C	ε	σ
20	10.0	0.1	5.0
80	0. 1	10.0	1.0

Table 8 Average RMSE and MAE achieved by the quantum and classical kernels when trained with 20 samples in the 8-qubit example

Type	Kernel	RMSE	MAE
Quantum kernel	Untrained quantum kernel	0.4389	0.3638
	Trained quantum kernel	**0.3544**	**0.2852**
Classical kernel	Poly	0.6796	0.5614
	Linear	0.6199	0.5315
	RBF	**0.5667**	**0.4573**

The bold indicates the lowest RMSE and MAE values

Table 9 Average RMSE and MAE achieved by the quantum and classical kernels when trained with 80 samples in the 8-qubit example

Type	Kernel	RMSE	MAE
Quantum kernel	Untrained quantum kernel	0.3113	0.2315
	Trained quantum kernel	**0.2761**	**0.2098**
Classical kernel	Poly	0.4843	0.3462
	Linear	0.4913	0.3684
	RBF	**0.3460**	**0.2531**

The bold indicates the lowest RMSE and MAE values

note that the average values of RMSE and MAE in our proposed QKE-QSVR model (with the trained quantum kernel) are 0.3544 and 0.2852, respectively, while the minimum average values of RMSE and MAE obtained using the classical SVR model (with the RBF kernel) are 0.5667 and 0.4573. These findings indicate that our QKE-QSVR model significantly outperforms the classical SVR model.

Next, we will discuss the results obtained from experiments utilizing 80 training samples, as shown in Table 9. It is evident that the trained quantum kernel reduces the average value of RMSE by 11.31% and the average value of MAE by 9.37% compared to the untrained quantum kernel, further confirming the effectiveness of the QKAR algorithm. Furthermore, upon comparing the QKE-QSVR model (with the trained quantum kernel) with the classical SVR model, we found that the QKE-QSVR model achieves an average RMSE and MAE of 0.2761 and 0.2098, respectively, while the minimum average values of RMSE and MAE obtained using the classical SVR model (with the RBF kernel) are 0.3460 and 0.2531, respectively. These results provide

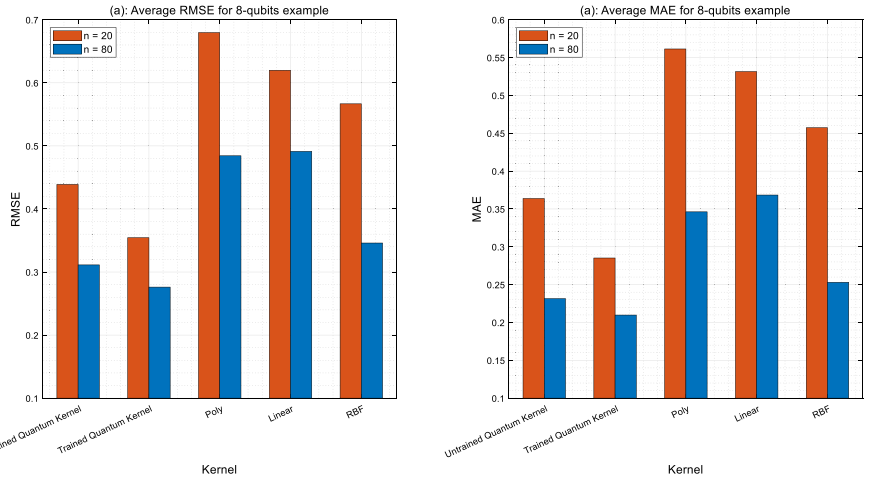

Fig. 4 Average RMSE (**a**) and MAE (**b**) achieved by quantum and classical kernels using 20 (red) and 80 (blue) training samples in the 8-qubit example (Color figure online)

compelling evidence of the superior predictive accuracy of the proposed QKE-QSVR model when compared to the classical SVR model.

Upon comparing the results obtained from models trained with 20 and 80 samples (Fig. 4), it is found that the prediction accuracy of the QKE-QSVR model is higher than that of the classical SVR model, regardless of whether 20 or 80 training samples are used. Additionally, Fig. 4 shows that the QKE-QSVR model trained with 80 training samples has a higher predictive accuracy and is more representative. These findings are approximately identical to those in the 2-qubit example.

4.3 A real-life example

In the previous sections, we have shown the effectiveness of the proposed QKE-QSVR model in handling datasets generated by two test functions. In order to further verify the model's capability to deal with real-life data, we apply it to the NASA airfoil self-noise dataset [45], which contains test results of NACA 0012 airfoils (n0012-il) of different sizes at varying wind tunnel speeds and angles of attack. As observed in the previous examples, the QKE-QSVR and SVR models trained with 80 training samples exhibit higher predictive accuracy and are more representative. Hence, in this example, we randomly select 80 training samples and 20 testing samples from the NASA airfoil self-noise dataset to validate the performance of the proposed QKE-QSVR model on a real-life dataset.

Similar to the workflow employed in the previous experiments, the first step is to select a QFM that offers the best performance for the given dataset. Its corresponding quantum kernel, namely the untrained quantum kernel, is then utilized in a classical SVR algorithm. Following this, a variational parameter is introduced in the QFM circuit to obtain a trainable quantum kernel, which is trained using the QKAR algorithm. The trained quantum kernel is then applied in a classical SVR procedure to

Table 10 The hyperparameter values for the QKE-QSVR model and the classical SVR model in the real-life example

Model	Hyperparameter		
	C	ε	σ
QKE-QSVR	10.0	0.1	
SVR	10.0	0.1	5.0

Table 11 Average RMSE and MAE achieved by quantum and classical kernels in the real-life example

Type	Kernel	RMSE	MAE
Quantum kernel	Untrained quantum kernel	0.3036	0.2389
	Trained quantum kernel	**0.2744**	**0.2199**
Classical kernel	Poly	0.3549	0.2818
	Linear	0.4007	0.3127
	RBF	**0.3266**	**0.2609**

The bold indicates the lowest RMSE and MAE values

find the optimal regression function. Meanwhile, we compare the performance of the proposed QKE-QSVR model with the classical SVR model based on different traditional kernel functions. Consistent with previous examples, the hyperparameters for the QKE-QSVR and the classical SVR model are adjusted through cross-validation. Table 10 demonstrates the specific hyperparameter values for the QKE-QSVR model and the classical SVR model in this example.

As with the previous experiments, the above process is repeated ten times, and the average results are considered as the final outcomes, which are shown in Table 11 and Fig. 5.

It can be seen that the trained quantum kernel outperforms the untrained one. Specifically, the trained quantum kernel reduces the average value of RMSE by 9.62% and the average value of MAE by 7.95% compared to the untrained quantum kernel. This indicates that the QKAR algorithm can effectively improve the predictive power of the model on this real-life dataset. Additionally, we observe that the proposed QKE-QSVR model exhibits better performance than the classical SVR model on this real-life dataset. Notably, the average values of RMSE and MAE achieved by the QKE-QSVR model (with the trained quantum kernel) are 0.2744 and 0.2199, whereas the minimum average values of RMSE and MAE achieved by the classical SVR model (with the RBF kernel) are 0.3266 and 0.2609, respectively. These findings suggest that the proposed QKE-QSVR model can achieve quantum advantage on this real dataset.

In a nutshell, it can be observed from the results of the above-mentioned experiments, the predictive accuracy of the proposed QKE-QSVR model is superior to the classical SVR model on both analogue and real datasets. This provides evidence for the strong predictive ability of our QKE-QSVR model in dealing with the regression task.

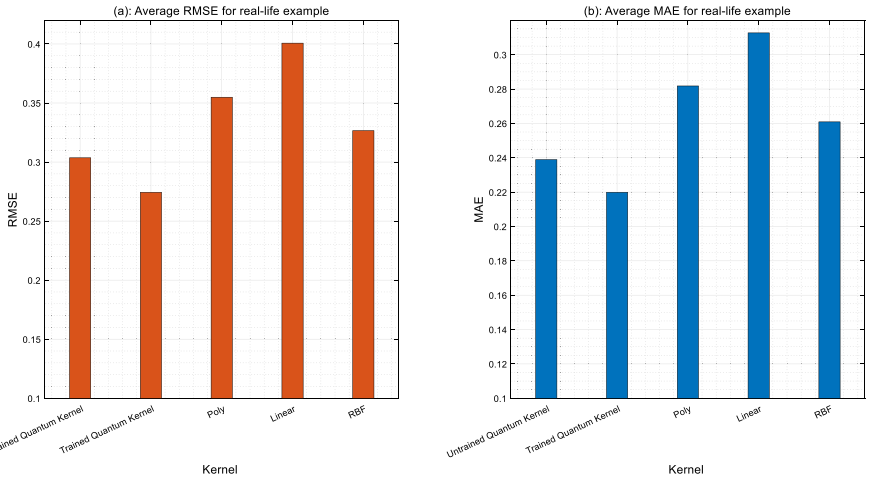

Fig. 5 Average results of RMSE (**a**) and MAE (**b**) achieved by quantum and classical kernels in the real-life example

5 Conclusion and outlook

In this paper, we propose a QKE-QSVR model to extend the application of QKE in regression tasks. The essence of the proposed QKE-QSVR involves employing quantum feature map to transform classical inputs into quantum states within the quantum Hilbert space. The computation of the quantum kernel is achieved by evaluating the transition probability amplitude between different quantum states. The exponential dimensionality of the quantum Hilbert space allows metrics in this space to more accurately represent the similarities among data points in the original space, leading to improved fitting and predictive accuracy. Furthermore, to enhance the adaptability of the proposed model, we introduce a trainable parameter within the quantum circuit and propose a QKAR algorithm to optimize it, thereby ensuring a robust generalization capability for a given dataset. We substantiate the proposed model's efficacy using three examples. Our findings indicate that the proposed QKE-QSVR model outperforms the classical SVR model in terms of predictive accuracy on all three examples. Our work not only furnishes a potentially novel method with high predictive accuracy for regression tasks in large-scale datasets but also contributes to the extension of the theoretical framework and application domains of the QKE method. Concurrently, the suggested framework exhibits certain scalability. In the future, further investigation into its performance on larger datasets and more complex regression tasks can be explored.

In this study, we have validated the performance of the proposed QKE-QSVR model on the Qiskit's Statevector Simulator. However, recent NISQ computers suffer from unavoidable device noise and can only perform a limited number of measurements, which may impact the model performance to some extent. Hence, future research will concentrate on verifying the performance of our proposed model on the NISQ computers and investigating how to mitigate device noise and statistical fluctuations. To be

specific, one might consider exploring noise-suppression techniques, quantum error correction techniques, and quantum circuits resilient to noise. Moreover, in the future, the presented QKE-QSVR model could find applications in areas such as investment risk prediction and industrial engineering design. For instance, one could contemplate employing the QKE-QSVR model as a response surface for robust parameter design (RPD) to address issues related to the slow construction speed and limited fitting accuracy of classical response surfaces.

Acknowledgements The funding provided for this study by the National Natural Science Foundation of China under Grant No.71872088, 71904078 and 71401080, the Social Science Foundation of Jiangsu under Grant No.17GLB016, Postgraduate Research & Practice Innovation Program of Jiangsu Province (Grant number: KYCX22_0881 and KYCX23_0934), the State Scholarship Fund of China under Grant No.201508320059, 1311 Talent Fund of NJUPT, the Science Foundation of Jiangsu under Grant No.BK20190793, the Project of Philosophy and Social Science Research in Colleges and Universities in Jiangsu Province under Grant No. 2018SJA0263 and Social Science Foundation of NJUPT under Grant No.NY218064 are gratefully acknowledged.

Data availability The datasets analyzed during the current study can be obtained from the corresponding author upon reasonable request.

Declarations

Conflict of interest All authors declare that they have no conflicts of interest.

References

1. Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., Lloyd, S.: Quantum machine learning. Nature **549**(7671), 195–202 (2017)
2. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev. **41**(2), 303–332 (1999)
3. Grover, L. K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
4. Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Phys. Rev. Lett. **103**(15), 150502 (2009)
5. Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum support vector machine for big data classification. Phys. Rev. Lett. **113**(13), 130503 (2014)
6. Liu, Y., Arunachalam, S., Temme, K.: A rigorous and robust quantum speed-up in supervised machine learning. Nat. Phys. **17**(9), 1013–1017 (2021)
7. Gao, X., Zhang, Z.-Y., Duan, L.-M.: A quantum machine learning algorithm based on generative models. Sci. Adv. **4**(12), eaat9004 (2018)
8. Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A., Woerner, S.: The power of quantum neural networks. Nat. Comput. Sci. **1**(6), 403–409 (2021)
9. Du, Y., Hsieh, M.-H., Liu, T., Tao, D.: Expressive power of parametrized quantum circuits. Phys. Rev. Res. **2**(3), 033125 (2020)
10. Wright, L.G., McMahon, P.L.: The Capacity of Quantum Neural Networks. p. JM4G. 5, CLEO: Science and Innovations: Optica Publishing Group (2020)
11. Jäger, J., Krems, R.V.: Universal expressiveness of variational quantum classifiers and quantum kernels for support vector machines. Nat. Commun. **14**(1), 576 (2023)
12. Caro, M.C., Huang, H.-Y., Cerezo, M., Sharma, K., Sornborger, A., Cincio, L., et al.: Generalization in quantum machine learning from few training data. Nat. Commun. **13**(1), 1–11 (2022)
13. Devitt, S.J., Stephens, A.M., Munro, W.J., Nemoto, K.: Requirements for fault-tolerant factoring on an atom-optics quantum computer. Nat. Commun. **4**(1), 2524 (2013)
14. Roetteler, M., Naehrig, M., Svore, K. M., Lauter, K.: Quantum resource estimates for computing elliptic curve discrete logarithms. Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference

on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3–7, 2017, Proceedings, Part II 23: Springer; 2017. pp. 241–270
15. Preskill, J.: Quantum computing in the NISQ era and beyond. Quantum **2**, 79 (2018)
16. Leymann, F., Barzen, J.: The bitter truth about gate-based quantum algorithms in the NISQ era. Quantum Sci. Technol. **5**(4), 044007 (2020)
17. Park, J.-E., Quanz, B., Wood, S., Higgins, H., Harishankar, R.: Practical application improvement to Quantum SVM: theory to practice. arXiv preprint arXiv:201207725. (2020)
18. Wu, S.L., Sun, S., Guan, W., Zhou, C., Chan, J., Cheng, C.L., et al.: Application of quantum machine learning using the quantum kernel algorithm on high energy physics analysis at the LHC. Phys. Rev. Res. **3**(3), 033221 (2021)
19. Hubregtsen, T., Wierichs, D., Gil-Fuster, E., Derks, P.-J.H., Faehrmann, P.K., Meyer, J.J.: Training quantum embedding kernels on near-term quantum computers. Phys. Rev. A **106**(4), 042431 (2022)
20. Nohara, T., Oyama, S., Noda, I.: Pairwise classification using quantum support vector machine with Kronecker kernel. Quantum Mach. Intell. **4**(2), 22 (2022)
21. Heyraud, V., Li, Z., Denis, Z., Le Boité, A., Ciuti, C.: Noisy quantum kernel machines. Phys. Rev. A **106**(5), 052421 (2022)
22. Torabian, E., Krems, R.V.: Compositional optimization of quantum circuits for quantum kernels of support vector machines. Phys. Rev. Res. **5**(1), 013211 (2023)
23. Gambetta, J.: IBM's roadmap for scaling quantum technology. IBM Research Blog (September 2020). (2020)
24. Sivak, V., Eickbusch, A., Royer, B., Singh, S., Tsioutsios, I., Ganjam, S., et al.: Real-time quantum error correction beyond break-even. Nature **616**(7955), 50–55 (2023)
25. Krinner, S., Lacroix, N., Remm, A., Di Paolo, A., Genois, E., Leroux, C., et al.: Realizing repeated quantum error correction in a distance-three surface code. Nature **605**(7911), 669–674 (2022)
26. Lloyd, S., Schuld, M., Ijaz, A., Izaac, J., Killoran, N.: Quantum embeddings for machine learning. arXiv preprint arXiv:200103622. (2020)
27. Havlíček, V., Córcoles, A.D., Temme, K., Harrow, A.W., Kandala, A., Chow, J.M., et al.: Supervised learning with quantum-enhanced feature spaces. Nature **567**(7747), 209–212 (2019)
28. Huang, H.-Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., et al.: Power of data in quantum machine learning. Nat. Commun. **12**(1), 1–9 (2021)
29. De Luca, G.: A survey of NISQ era hybrid quantum-classical machine learning research. J. Artif. Intell. Technol. **2**(1), 9–15 (2022)
30. McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. Nat. Commun. **9**(1), 4812 (2018)
31. Holmes, Z., Sharma, K., Cerezo, M., Coles, P.J.: Connecting ansatz expressibility to gradient magnitudes and barren plateaus. PRX Quantum **3**(1), 010313 (2022)
32. Cerezo, M., Sone, A., Volkoff, T., Cincio, L., Coles, P.J.: Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nat. Commun. **12**(1), 1791 (2021)
33. Patti, T.L., Najafi, K., Gao, X., Yelin, S.F.: Entanglement devised barren plateau mitigation. Phys. Rev. Res. **3**(3), 033090 (2021)
34. Cincio, L., Subaşı, Y., Sornborger, A.T., Coles, P.J.: Learning the quantum algorithm for state overlap. New J. Phys. **20**(11), 113022 (2018)
35. Schuld, M.: Supervised quantum machine learning models are kernel methods. arXiv preprint arXiv:210111020. (2021)
36. Srikumar, M., Hill, C. D., Hollenberg, L. C.: A kernel-based quantum random forest for improved classification. arXiv preprint arXiv:221002355. (2022)
37. Paine, A.E., Elfving, V.E., Kyriienko, O.: Quantum kernel methods for solving regression problems and differential equations. Phys. Rev. A **107**(3), 032428 (2023)
38. Sancho-Lorente, T., Román-Roche, J., Zueco, D.: Quantum kernels to learn the phases of quantum matter. Phys. Rev. A **105**(4), 042432 (2022)
39. Tancara, D., Dinani, H.T., Norambuena, A., Fanchini, F.F., Coto, R.: Kernel-based quantum regressor models learning non-Markovianity. Phys. Rev. A **107**(2), 022402 (2023)
40. Fadol, A., Sha, Q., Fang, Y., Li, Z., Qian, S., Xiao, Y., et al.: Application of Quantum Machine Learning in a Higgs Physics Study at the CEPC. arXiv preprint arXiv:220912788. (2022)
41. Mensa, S., Sahin, E., Tacchino, F., Barkoutsos, P. K., Tavernelli, I.: Quantum Machine Learning Framework for Virtual Screening in Drug Discovery: A Prospective Quantum Advantage. arXiv preprint arXiv:220404017. (2022)

42. Gawriljuk, V.O., Zin, P.P.K., Puhl, A.C., Zorn, K.M., Foil, D.H., Lane, T.R., et al.: Machine learning models identify inhibitors of SARS-CoV-2. J. Chem. Inf. Model. **61**(9), 4224–4235 (2021)
43. Adorio, E. P., Diliman, U.: Mvf-multivariate test functions library in c for unconstrained global optimization. Quezon City, Metro Manila, Philippines. 44. (2005)
44. Molga, M., Smutnicki, C.: Test functions for optimization needs. Test Funct. Optim. Needs **101**, 48 (2005)
45. Brooks, T. F., Pope, D. S., Marcolini, M. A.: Airfoil self-noise and prediction. (1989)
46. Benedetti, M., Lloyd, E., Sack, S., Fiorentini, M.: Parameterized quantum circuits as machine learning models. Quantum Sci. Technol. **4**(4), 043001 (2019)
47. Van Beers, W.C., Kleijnen, J.P.: Kriging for interpolation in random simulation. J. Oper. Res. Soc. **54**, 255–262 (2003)
48. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
49. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Stat. Comput. **14**(3), 199–222 (2004)
50. Awad, M., Khanna, R., Awad, M., Khanna, R.: Support vector regression. Efficient learning machines: Theories, concepts, and applications for engineers and system designers. 67–80 (2015)
51. Schölkopf, B., Smola, A. J., Bach, F.: Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press. (2002)
52. Kashima, H., Idé, T., Kato, T., Sugiyama, M.: Recent advances and trends in large-scale kernel methods. IEICE Trans. Inf. Syst. **92**(7), 1338–1353 (2009)
53. Thanasilp, S., Wang, S., Cerezo, M., Holmes, Z.: Exponential concentration and untrainability in quantum kernel methods. arXiv preprint arXiv:220811060. (2022)
54. Larocca, M., Sauvage, F., Sbahi, F.M., Verdon, G., Coles, P.J., Cerezo, M.: Group-invariant quantum machine learning. PRX. Quantum **3**(3), 030341 (2022)
55. Skolik, A., Cattelan, M., Yarkoni, S., Bäck, T., Dunjko, V.: Equivariant quantum circuits for learning on weighted graphs. NPJ Quantum Inf. **9**(1), 47 (2023)
56. Meyer, J.J., Mularski, M., Gil-Fuster, E., Mele, A.A., Arzani, F., Wilms, A., et al.: Exploiting symmetry in variational quantum machine learning. PRX Quantum. **4**(1), 010328 (2023)
57. Glick, J. R., Gujarati, T. P., Corcoles, A. D., Kim, Y., Kandala, A., Gambetta, J. M., et al.: Covariant quantum kernels for data with group structure. arXiv preprint arXiv:210503406. (2021)
58. Cortes, C., Mohri, M., Rostamizadeh, A.: Algorithms for learning kernels based on centered alignment. J. Mach. Learn. Res. **13**(1), 795–828 (2012)
59. Bullins, B., Zhang, C., Zhang, Y.: Not-so-random features. arXiv preprint arXiv:171010230. (2017)
60. Shawe-Taylor, J., Cristianini, N.: On the generalization of soft margin algorithms. IEEE Trans. Inf. Theory **48**(10), 2721–2735 (2002)
61. Spall, J.C.: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans. Autom. Control **37**(3), 332–341 (1992)
62. Pellow-Jarman, A., Sinayskiy, I., Pillay, A., Petruccione, F.: A comparison of various classical optimizers for a variational quantum linear solver. Quantum Inf. Process. **20**(6), 202 (2021)
63. Aleksandrowicz, G., Alexander, T., Barkoutsos, P., Bello, L., Ben-Haim, Y., Bucher, D., et al.: Qiskit: An open-source framework for quantum computing. Accessed on Mar 16 (2019)
64. Suzuki, Y., Yano, H., Gao, Q., Uno, S., Tanaka, T., Akiyama, M., et al.: Analysis and synthesis of feature map for kernel-based quantum classifier. Quantum Mach. Intell. **2**, 1–9 (2020)
65. Schuld, M., Killoran, N.: Quantum machine learning in feature Hilbert spaces. Phys. Rev. Lett. **122**(4), 040504 (2019)
66. Krunic, Z., Flöther, F.F., Seegan, G., Earnest-Noble, N.D., Shehab, O.: Quantum kernels for real-world predictions based on electronic health records. IEEE Trans. Quantum Eng. **3**, 1–11 (2022)
67. Li, W., Deng, D.-L.: Recent advances for quantum classifiers. Sci. China Phys. Mech. Astron. **65**(2), 220301 (2022)