scientific reports

OPEN

The power of one clean qubit in supervised machine learning

Mahsa Karimi[1,2✉], Ali Javadi-Abhari[3], Christoph Simon[1,2] & Roohollah Ghobadi[1,2✉]

This paper explores the potential benefits of quantum coherence and quantum discord in the non-universal quantum computing model called deterministic quantum computing with one qubit (DQC1) in supervised machine learning. We show that the DQC1 model can be leveraged to develop an efficient method for estimating complex kernel functions. We demonstrate a simple relationship between coherence consumption and the kernel function, a crucial element in machine learning. The paper presents an implementation of a binary classification problem on IBM hardware using the DQC1 model and analyzes the impact of quantum coherence and hardware noise. The advantage of our proposal lies in its utilization of quantum discord, which is more resilient to noise than entanglement.

Recent progress in the control and mitigation of noise and decoherence has paved the way for the development of intermediate-scale quantum devices consisting of hundreds of qubits. Although these devices are currently not fault-tolerant, there is considerable evidence that they possess superior computational capabilities compared to classical supercomputers, as a result of their ability to support quantum entanglement[1,2]. As quantum hardware continues to evolve, it is expected to play a crucial role in various fields such as quantum simulations, quantum chemistry, and quantum machine learning (QML)[3,4].

The use of quantum hardware for complex computations such as kernel function estimation has been proposed as a way to achieve a quantum advantage in machine learning[5,6]. Quantum entanglement is considered a key resource for this[7–9], but it is highly susceptible to noise, thus it is important to explore other forms of quantum correlation that are less sensitive to noise or require less entanglement.

The Deterministic Quantum Computing with One Qubit (DQC1) model is a non-universal quantum computing model that leverages a single qubit as a probe to interact with a highly mixed quantum state and estimate computationally expensive functions. This ability is known as the "power of one qubit"[10]. The DQC1 model generates quantum discord, a resilient type of weak quantum correlation, using the coherence of a pure qubit[11,12]. Quantum discord is more resistant to noise than entanglement and may offer a quantum advantage in noisy conditions for quantum illumination tasks[13].

There is a limited body of literature exploring the use of DQC1 in machine learning contexts[14–16]. Reference[14] investigates the advantages of DQC1 in addressing the parity learning problem. In reference[15], DQC1 is proposed for application in kernel based supervised machine learning. Finally, reference[16] builds upon the results in reference[15], extending the concept to multiple kernel learning for supervised machine learning within a DQC1 framework.

This paper studies the use of the DQC1 model in supervised machine learning for efficient estimation of complex kernel functions. The study is implemented on IBM hardware and examines the effects of coherence consumption, quantum discord, and hardware noise. The DQC1 protocol reduces measurement errors by only measuring one qubit, achieving high classification accuracy despite requiring more gates than a similar protocol in[6].

The paper is structured as follows: "Preliminaries" provides a review of the DQC1 algorithm, quantum coherence and quantum discord, and a brief overview of kernel-based supervised machine learning. In "Supervised machine learning with DQC1", we describe the application of DQC1 for the estimation of arbitrary kernel functions. "Implementation on IBM hardware" presents our implementation of supervised machine learning using DQC1 on IBM hardware. We also compare the DQC1 kernel with the projected quantum kernel in this section. "The role of coherence and the effect of noise" presents the role of quantum coherence and the effect of noise in our implementation. Finally, "Discussion and conclusion" discusses the limitations imposed on the DQC1 kernel and concludes with a summary of our findings.

[1]Department of Physics and Astronomy, University of Calgary, Calgary, AB T2N 1N4, Canada. [2]Institute for Quantum Science and Technology, University of Calgary, Calgary, AB T2N 1N4, Canada. [3]IBM Quantum, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA. ✉email: mahsa.karimi1@ucalgary.ca; rghobadi@ucalgary.ca

Preliminaries

DQC1

The DQC1 model was originally introduced in the context of nuclear magnetic resonance (NMR) quantum information processing and has been implemented in various physical setting[17–20].

As shown in Fig. 1, the DQC1 circuit consists of one control qubit which is prepared in $\frac{I_1+\alpha Z}{2}$ with $\alpha \in [0, 1]$, Z as Pauli Z matrix and n target qubits in a maximally mixed state denoted by $\frac{I_n}{2^n}$ where I_n is a $2^n \times 2^n$ identity matrix. One can change the purity of the control qubit by tuning $\alpha \in [0, 1]$: for $\alpha = 0$ and $\alpha = 1$ the control qubit will be in maximal mixed and pure states, respectively, while for $0 < \alpha < 1$ the control qubit will be in a partially mixed state. Once the control qubit is evolved through the Hadamard gate, as shown in Fig. 1, the matrix form of the initial state in the computational basis of control qubit becomes

$$\rho_{in} = \frac{1}{2^{n+1}} \begin{pmatrix} I_n & \alpha I_n \\ \alpha I_n & I_n \end{pmatrix}. \tag{1}$$

Following the application of the DQC1 circuit evolution

$$U_{DQC1} = |0\rangle\langle 0| \otimes I_n + |1\rangle\langle 1| \otimes U_n, \tag{2}$$

where U_n is an arbitrary $2^n \times 2^n$ unitary matrix that is applied to n target qubits, the total state is updated to

$$\rho_f = U_{DQC1} \rho_{in} U_{DQC1}^\dagger = \frac{1}{2^{n+1}} \begin{pmatrix} I_n & \alpha U_n^\dagger \\ \alpha U_n & I_n \end{pmatrix}. \tag{3}$$

Tracing out the last n qubits from Eq. (3), the density matrix of the control qubit denoted by $\rho_{f,c}$ is given by

$$\rho_{f,c} = \frac{1}{2} \begin{pmatrix} 1 & \frac{\alpha}{2^n}\text{tr}(U_n^\dagger) \\ \frac{\alpha}{2^n}\text{tr}(U_n) & 1 \end{pmatrix}. \tag{4}$$

Measuring the off-diagonal elements of the control qubit can be used to calculate the trace of a unitary matrix, as demonstrated in Eq. (4). The number of measurements needed to estimate the off diagonal elements in Eq. (4) within precision ϵ and with probability $1 - \delta$ is $O(\epsilon^{-2}\alpha^{-2}\log(1/\delta))$, which is independent of the number of register qubits. Therefore, DQC1 serves as an efficient method for estimating the trace of an arbitrary unitary matrix, a problem for which no efficient classical algorithm is known[10]. In the special case of a real positive semi-definite matrix, there is a classical randomized algorithm to estimate the trace[21]. As pointed out in[22], the capability of DQC1 to achieve universal classical computation is still an open question. However, there exist complexity arguments that prove the classical efficient approximation of the output probability distribution of the DQC1 model is impossible unless the polynomial-time hierarchy collapses to the second level[23].

For completeness, we will now demonstrate that the clean qubit and register qubits remain in a separable state throughout the computation as stated in[24]. To do this, we will use the eigenvectors and eigenvalues of U_n, denoted as $|u_i\rangle$ and λ_i, respectively. In the basis of $\{|u_i\rangle\}$, the mixed state of the target qubits can be represented as $I_n = \sum_i |u_i\rangle\langle u_i|$. Applying the DQC1 evolution as defined in Eq. (2) results in

$$U_{DQC1}(|0\rangle + |1\rangle)|u_i\rangle = (|0\rangle + \lambda_i|1\rangle)|u_i\rangle, \tag{5}$$

which is clearly a product state, meaning there is no entanglement between the control qubit and the target qubits.

DQC1 resource

The DQC1 model has been widely studied to investigate the potential use of quantum resources, other than entanglement, in quantum computation[25,26]. In this section, we shortly review the definitions of quantum coherence and discord and we demonstrate that it is the consumption of this coherence that allows for the production of discord[27].

The rigorous definition of coherence was first given in[28]. The coherence is defined as

$$C(\rho) = S(\rho_{\text{diag}}) - S(\rho), \tag{6}$$

where $S(\rho) = -\text{tr}(\rho\log\rho)$ is the von Neumann entropy and ρ_{diag} is the diagonal part of ρ. One can find the change of coherence for the control qubit defined as $\Delta C = C(\rho_{in,c}) - C(\rho_{f,c})$ with $\rho_{in,c}$ and $\rho_{f,c}$ as input and output state for the control qubit, respectively. The input state of the control qubit can be obtained by tracing out the Eq. (1) (See Eq. S1 in supplementary material). Using Eq. S1 and Eq. (4) in Eq. (6) one obtains[14]
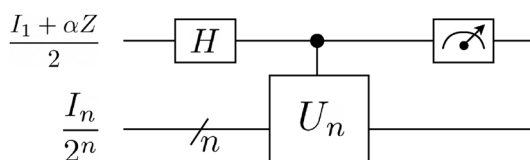

Figure 1. The circuit representation of the DQC1 algorithm. The input states for control and target qubits are $\frac{I_1+\alpha Z}{2}$, with $\alpha \in [0, 1]$ and $\frac{I_n}{2^n}$, respectively. H and Z denote the Hadamard and Pauli Z gates, respectively.

$$\Delta C = H_2\left(\frac{1 - \alpha\frac{|\text{tr}(U_n)|}{2^n}}{2}\right) - H_2\left(\frac{1 - \alpha}{2}\right), \tag{7}$$

where $H_2(x) = -x\log_2 x - (1-x)\log_2(1-x)$ is the binary Shannon entropy (see Supplementary material for the derivation). From Eq. (7), it is clear that the coherence consumption which is determined by the parameter α and the trace of U_n, can be obtained efficiently by DQC1.

Quantum discord is a generalization of the classical notion of mutual information and is defined as the difference between the total quantum mutual information and the classical mutual information of the subsystems. For a bipartite system in state ρ_{AB}, the quantum discord is defined by the difference

$$D(\rho_{AB}) = I(\rho_{AB}) - J(\rho_{AB}), \tag{8}$$

with $I(\rho_{AB})$ and $J(\rho_{AB})$ as the quantum mutual information, and the measurement-based mutual information, respectively. The quantum mutual information is given by

$$I(\rho_{AB}) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}). \tag{9}$$

The measurement-based mutual information on the other hand is given by

$$J(\rho_{AB}) = S(\rho_B) - \min_{E_k}[p_k S(\rho_{B|k})], \tag{10}$$

where the minimum is taken over all possible positive operator-valued measurements (POVM) $\{E_k\}$ on subsystem A, $\rho_{B|k} = \text{tr}_A(E_k\rho_{AB})/p_k$ is the post-measurement state for system B if outcome k is obtained with probability $p_k = \text{tr}(E_k\rho_{AB})$.

In the following, we use an alternative definition for discord known as geometric discord, which is easier to calculate and has a closed form for DQC1[29]. For a given quantum state ρ, the geometric discord is defined as[30]

$$D_G(\rho) = \min_{\chi \in \mathcal{C}}||\rho - \chi||^2, \tag{11}$$

where $\mathcal{C}$ denotes the set of classical zero-discord states and $||A - B||^2 = \text{tr}(A - B)^2$. Evaluating Eq. (11) for the state in Eq. (3) leads to[29]

$$D_G(\rho_f) = \left(\frac{\alpha}{2}\right)^2 \frac{1}{2^n}\left(1 - \frac{|\text{tr}(U_n^2)|}{2^n}\right). \tag{12}$$

Evaluating Eq. (12) can be done by applying two consecutive controlled-U_n in Fig. 1. In[27], the connection between coherence consumption and discord production in DQC1 was examined, where it was demonstrated that quantum discord is bounded by the quantum coherence consumed in the control qubit, i.e.

$$D_G(\rho_f) \leq \Delta C. \tag{13}$$

In our implementation, the results obtained from IBM hardware verifies the relation (13).

Supervised machine learning: support vector machines and kernel method

In this section, we introduce the concepts of support vector machines (SVM) and the kernel method within the context of supervised machine learning. Given a set of n training data points, represented by $X_{\text{train}} := \{(x_i, y_i) : i = 1, 2, ..., n\}$, where each data point has k features, i.e., $x_i \in \mathbb{R}^k$, and is labeled by $y_i \in \{1, -1\}$. The task is to use the training data to develop a classifier function that can accurately predict the labels for test (unseen) data. In the simplest scenario, where the data points are linearly separable, the classifier function can be expressed as

$$f(x) = \text{sign}(w^T x + b), \tag{14}$$

where $w \in \mathbb{R}^k$ and b are to be determined such that $y_i f(x_i) > 0$. In the SVM, the separating plane $f(x)$ is determined by maximizing the distance between the hyperplane to the nearest data point of each class[31], see Fig. 2a.

The SVM can be generalized to the case of non-linearly separated data points by mapping the data points to a higher dimension space for which the data is linearly separable, see Fig. 2b. In other words, one considers a non-linear mapping $\phi : X \to \mathcal{H}$, so that the decision function can be written as

$$f(x) = \text{sign}(w^T \phi(x) + b). \tag{15}$$

In this context, $\mathcal{H}$ and $\phi(x)$ are known as feature space and feature map, respectively.

It is well-known that for nonlinear separable data points the SVM leads to solutions of the form[32],

$$f(x) = \text{sign}\sum_i \beta_i^* K(x, x_i), \tag{16}$$

where β_i^* are coefficients to be determined, and we defined the kernel function $K(x, x_i) = \langle\phi(x), \phi(x_i)\rangle$, where $\langle,\rangle$ denotes the inner product in feature space $\mathcal{H}$. The procedure for finding β_i in Eq. (16) is through maximizing
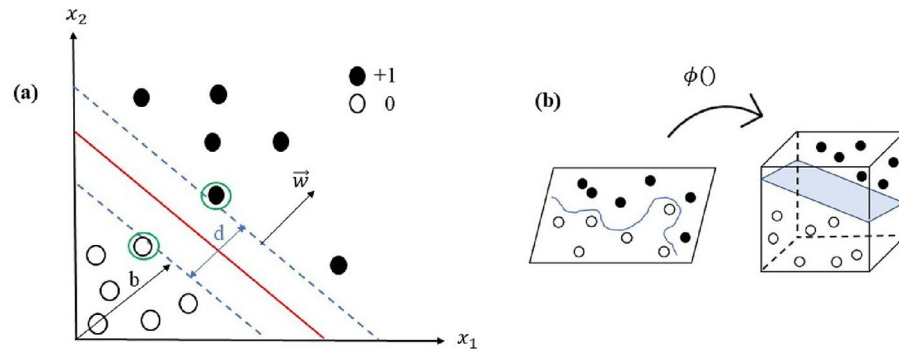
Figure 2. (a) A Support Vector Machine (SVM) is a classifier used to separate two linearly separable classes, depicted in black and white. The data points closest to the decision boundary (shown in red), one from each class, are known as support vectors and are indicated by green circles. (b) When data points of two classes cannot be separated by a hyperplane in the original space (left), a non-linear mapping can be applied to project the data points into a higher-dimensional feature space (right) where a hyperplane can be found to separate the classes.

$$\sum_{i=1}^{n} \beta_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \beta_i \beta_j K(x_i, x_j), \tag{17}$$

over the training data, subject to $\sum_{i=1}^{n} \beta_i y_i = 0$ and $\beta_i \geq 0$. For a positive definite kernel, Eq.(17) is a concave problem, whose solution $\beta^* = (\beta_1^*, ..., \beta_n^*)$ can be found efficiently.

The basic idea of SVM can be extended to the quantum domain by interpreting the feature map as a quantum state that can be constructed by a quantum circuit, and the kernel function as the inner product between respective quantum states[6,33].

Supervised machine learning with DQC1

The freedom in choosing the unitary operator U_n in DQC1 allows one to make a connection between DQC1 and the kernel method[15]. To see this, we choose $U_n = u^l(\mathbf{x})\left(u^l(\mathbf{x}')\right)^\dagger$, where u^l represents l consecutive application of unitary operator u with $\mathbf{x}$ and $\mathbf{x}'$ as encoded data points in the gate parameters. Next, we note that $\mathrm{tr}(u^l(\mathbf{x})\left(u^l(\mathbf{x}')\right)^\dagger)$ is positive semidefinite, i.e. $\sum_{i,j} c_i c_j \mathrm{tr}(u^l(\mathbf{x})\left(u^l(\mathbf{x}')\right)^\dagger) \geq 0$ for $c_i \in \mathbb{C}$.

Rewriting Eq. (4) for $U_n(\mathbf{x}, \mathbf{x}') = u^l(\mathbf{x})\left(u^l(\mathbf{x}')\right)^\dagger$,

$$\rho_{f,c} = \frac{1}{2} \begin{pmatrix} 1 & \alpha K^*(\mathbf{x}, \mathbf{x}') \\ \alpha K(\mathbf{x}, \mathbf{x}') & 1 \end{pmatrix}, \tag{18}$$

where $K(\mathbf{x}, \mathbf{x}') = \frac{\mathrm{tr}(U_n(\mathbf{x}, \mathbf{x}'))}{2^n}$.

From Eq. (18), it follows that the DQC1 model allows for an efficient method for estimating arbitrary, complicated kernel functions.

Interestingly, by comparing Eq. (18) with equation (7), we can relate the coherence consumption to the kernel function. For example, by setting $\alpha = 1$ in equation (7), we obtain

$$\Delta C(\mathbf{x}, \mathbf{x}') = H_2\left(\frac{1 - |K(\mathbf{x}, \mathbf{x}')|}{2}\right). \tag{19}$$

From Eq. (19) the following key insight can be obtained. Firstly, when $\Delta C(\mathbf{x}, \mathbf{x}') = 0$ it follows that $K(\mathbf{x}, \mathbf{x}') = 1$, indicating that the kernel function is incapable of distinguishing between the two data points. This lack of discrimination hinders the learning process and underscores the significance of coherence consumption in learning. Secondly, Eq. (19) provides a means to observe the influence of hardware noise. To see this, note that in the absence of hardware noise, one expects that $\Delta C(\mathbf{x}, \mathbf{x}) = 0$ as $K(\mathbf{x}, \mathbf{x}) = 1$. In real situations, where noise cannot be ignored, the diagonal elements of a kernel will be smaller than one, related to a loss of coherence as shown by Eq. (19) (see "The role of coherence and the effect of noise" for more details.). Please note that it follows from Eq. (7) that the above conclusions are applicable for arbitrary $\alpha \in (0, 1]$.

Implementation on IBM hardware

In this section, we describe our implementation of supervised machine learning based on the DQC1 model. Our scheme was implemented on the "*ibm_perth*" quantum processor, as shown in Fig. 3, using IBM's open-source software interface, Qiskit.

Our demonstration were performed on ibm_perth, which is a $7-$qubit superconducting quantum processor from IBM. It consists of fixed-frequency transmon qubits connected according to the coupling map of Fig. 3. The device characteristics at the time of our demonstration were as follows: Median $T1 : 137.24 \mu s$ Median
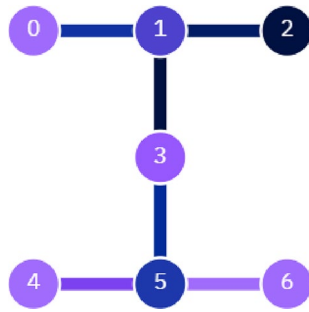
Figure 3. Architecture of 7-qubit "*ibm_perth*" quantum device.

$T2$: 107.32µs Median 1-qubit gate error: 0.03% Median 2-qubit gate error: 1.06% Median readout error: 4.56% The code was implemented on top of Qiskit Machine Learning, which provides the dataset and the SVM kernels. The code and Qiskit version for these demonstration can be found online[34].

Figure 4a shows the schematics of our implemented circuit which is composed of two target qubits and three ancilla qubits. In the first part of the circuit, left to the dashed line, input states for control and target qubits are
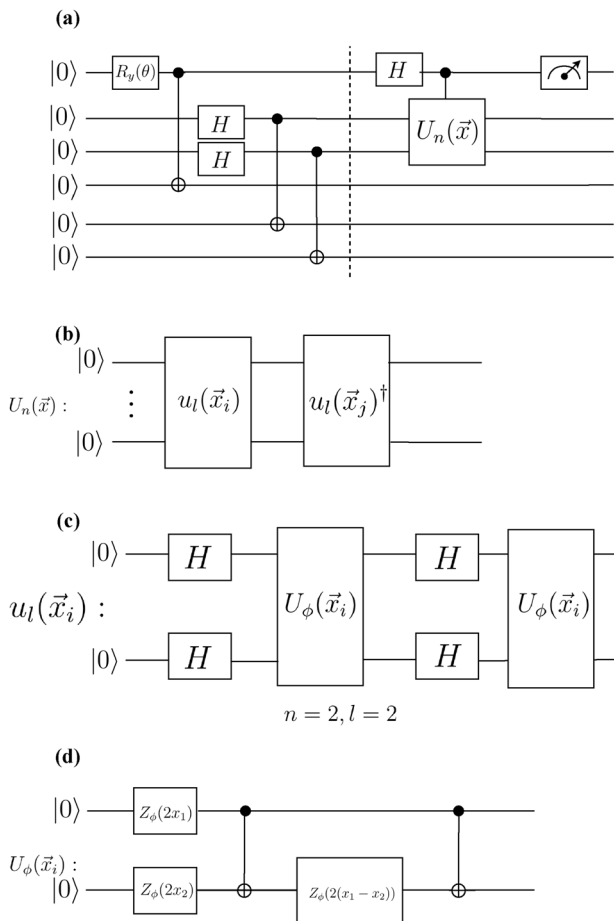
Figure 4. (**a**) A schematic picture of a three-qubit version of DQC1 circuit, with one control-qubit, and two target qubits and three ancilla qubits. The first part of the circuit, before the dashed line, prepares the control qubit in $\frac{I_1 + \alpha Z}{2}$ and target qubits in mixed state. Here, $R_y(\theta) = \exp(-i\frac{\theta}{2}Y)$, is a rotation gate around y axis, and Y denotes Pauli Y gate. (**b**) The gate decomposition for the unitary matrix $U_n = u^l(\mathbf{x}_i)\left(u^l(\mathbf{x}_j)\right)^\dagger$. $u^l(\mathbf{x}_i)$ is the encoding circuit, where l is the number of the iterations of this gate decomposition (length of the circuit). (**c**) The gate components of the unitary operator $u^l(\mathbf{x}_i)$ adapted from[6] for two target qubits $n = 2$, and two iterations $l = 2$. Here $U_\phi(\mathbf{x}_i)$ is the feature map (defined in the text). At first, the Hadamard gate is applied to all qubits and then a diagonal gate $U_\phi(\mathbf{x}_i)$ acts on the qubits. (**d**) The feature encoding circuit $U_\phi(\mathbf{x}_i)$ in Eq.(21). Here, $Z_\phi = \mathrm{diag}(1, e^{i\phi(\mathbf{x}_i)})$ is a single qubit phase gate.

prepared. The mixed state preparation of target qubits are based on creating Bell states between target and ancilla qubits, followed by ignoring the state of ancilla qubits. The resulting state of the control and target qubits right before the dashed line is given by $\rho_{control} \otimes \frac{I_2}{4}$ where $\rho_{control} = \text{diag}\left(\cos^2 \frac{\theta}{2}, \sin^2 \frac{\theta}{2}\right)$. By choosing $\theta = 2\cos^{-1}\left(\sqrt{\frac{1+\alpha}{2}}\right)$ the state of control qubit becomes $\frac{I_1 + \alpha Z}{2}$.

In our implementation on the IBM real hardware, the purity of the resulting target qubits is $\text{tr}(\rho_{target}^2) = 0.506$, which deviates from the ideal mixed state by 6×10^{-3}.

To benchmark the performance of our protocol, we use the encoding map and dataset used in[6]. In[6] the data points $\mathbf{x}$ and $\mathbf{x}'$ are mapped onto gate parameters of the unitary matrix $U_2(\mathbf{x}, \mathbf{x}') = u^l(\mathbf{x})\left(u^l(\mathbf{x}')\right)^\dagger$, where

$$u^l(\mathbf{x}) = \prod_{i=0}^{l}(U_{\phi(\mathbf{x})}H^{\otimes 2})_i, \tag{20}$$

with l as the number of iterations of each layer in the feature map (See Fig. 4b), encoding map $U_{\phi(\mathbf{x})}$, and $H^{\otimes 2}$ denotes two Hadamard gates acting on two qubits (see Fig.4c).

$$U_{\phi(\mathbf{x})} = \exp\left(\sum_{S \subseteq [n]} \phi_S(\mathbf{x}) \prod_{i \in S} Z_i\right), \tag{21}$$

where $\phi_i(\mathbf{x}) = x_i$, $\phi_{i,j}(\mathbf{x}) = (\pi - x_i)(\pi - x_j)$, and Z_i denotes Pauli Z gate (See Fig.4(d), Fig. 4c shows the quantum circuit that describes Eq.(20) for $l = 2$. We defined our kernel as $K(\mathbf{x}, \mathbf{x}') = \frac{|\text{tr}(U_2(\mathbf{x}, \mathbf{x}'))|}{4}$. It has been conjectured that approximation of the resulting kernel function for the encoding map Eq. (21) with $l = 2$ is hard classically, i.e. the resources required to perform it, increase at a non-polynomial rate with respect to number of qubits[6].

Our implementation is divided into three phases. In the first stage, we run the circuit in Fig. 4 for all pairs of training data to obtain the corresponding density matrix for the control qubit, using the quantum state tomography package in Qiskit with repeating each measurement 8000 times (shots), and therefore to obtain the corresponding kernel function. Having obtained the kernel function on the quantum hardware, we apply the classical SVM to obtain the optimal separating hyperplane, or equivalently β^* by applying Eq. (17). Finally, in the prediction phase, given test data $\mathbf{x}$, we run the DQC1 circuit to estimate the $K(\mathbf{x}, \mathbf{x}_i)$ for all $\mathbf{x}_i \in X_{\text{train}}$ and apply Eq. (16).

In Fig. 5, we display the results of applying the above procedure for the classification task on the "ad_hoc" dataset for the IBM simulator (Qiskit) (left) and IBM 7-qubit hardware (right) for $l = 2$ for the control qubit in the pure state, i.e. $\alpha = 1$[34]. From Fig. 5, it can be seen that the accuracy of the Qiskit simulator is 100%. On the other hand, the obtained accuracy on the hardware is 90%. The difference between the simulation and hardware performance can be attributed to the effects of hardware noise. It is worth noting that the circuits were optimized using the Qiskit compiler, specifically the Approximate Quantum Compilation method[35]. This method converts the entire circuit (excluding ancilla) to a 3-qubit unitary matrix, and then re-synthesizes it into a new circuit that approximates the matrix with 0.995 accuracy (synthesis fidelity). A higher synthesis fidelity uses more CNOT gates in the resulting circuit, which reduces approximation error but increases runtime noise. This method reduced the CNOT gate count of the circuit from 177 to 19.

It is worth noting the similarity between our approach and the projected kernel method introduced in[36], where both methods involve constructing the kernel function by measuring a subset of the relevant qubits. More explicitly, the projected kernel function is defined as[36]
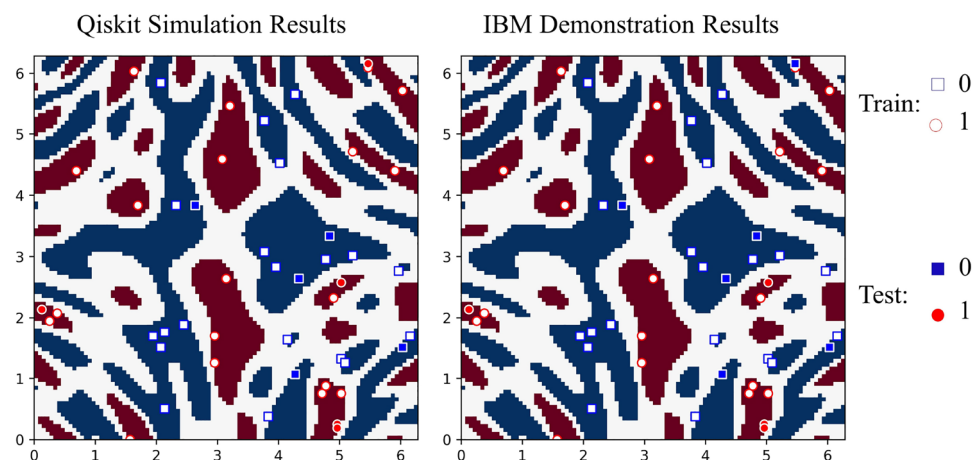

Figure 5. The Qiskit simulation (left) and IBM (right) results for the DQC1 kernel classification are shown in Fig.4 with $n = 2$ and $l = 2$. We used the "ad_hoc" dataset, which includes 20 training and 5 test samples per label. The accuracy of classification for the IBM quantum simulator (Qiskit) is 100%, while it is 90% for IBM's real hardware.

$$k^{PQ}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \sum_{m=1}^{n} \|\rho_m(\mathbf{x}_i) - \rho_m(\mathbf{x}_j)\|^2\right), \tag{22}$$

where $\rho_m(\mathbf{x}_i)$ is the reduced density matrix of the m-th register of the encoded quantum state $\rho(\mathbf{x}_i)$, and $\gamma > 0$ is a hyperparameter. Furthermore, their work extends this kernel to $k_s^{(PQ)}$ which takes every subset of s qubits into account (for clarification, we have $k_1^{(PQ)} = k^{(PQ)}$). We highlight that the number of measurements required in determining $k_s^{(PQ)}$ grows as $4^s \binom{n}{s}$ since there are 4^s Pauli strings on s qubits, and $\binom{n}{s}$ subsets of size s for a state of n qubits. This quantity grows polynomially in n and exponentially in s. In contrast, our method only necessitates the measurement of the control qubit. This practical aspect holds strong appeal because, in current superconducting qubit technology, measurements are the most error-prone operations, with error rates ranging from 3 to 10 times that of 2-qubit gates, as demonstrated in[37]. For completeness, we repeated our experiments employing the projected quantum kernel on "ad_hoc" dataset[34]. The two-qubit projected kernel is depicted in Fig. 6a with the same encoding circuit (Fig. 6b), and feature map (Fig. 6c) as in DQC1 kernel. The Qiskit simulation and IBM hardware's results (considering $\gamma = 0.01$) demonstrated an accuracy of 100% and 90%, respectively.

The role of coherence and the effect of noise

In the following, we explore the role of control qubit's coherence, hardware noise, coherence consumption and quantum discord production in our setting.

To see the role of the control qubit's coherence in our implementation, we repeat the learning task with the control qubit prepared in the state $\frac{I_1 + \alpha Z}{2}$, where $0 \leq \alpha \leq 1$. In Fig. 7 we show the prediction accuracy in the simulation (blue dots) and the implementation (red dots) as a function of the purity of the control qubit. From Fig. 7 one can see that $\alpha = 0$, for which the control qubit is in a maximally mixed state, the accuracy is 0.5, corresponding to randomly guessing the labels. By increasing the purity, however, the accuracy increases until it reaches its maximum value at $\alpha \geq 0.6$. Due to the device noise, the accuracy in the implementation is degraded in comparison to the simulation.

For completeness we repeat the learning process with two well-known datasets called "make-moon", and "make-circle" from "scikit-learn", each of them including 800 training data points, and 200 testing data points. For these two datasets we observed an abrupt change in the accuracy for $\alpha \geq 0.2$. Hence the critical value of α depends on the dataset. These results are depicted in Fig. 8. By interpreting $1 - \alpha$ as the noise strength, one can see that the accuracy is robust against noise for $\alpha \geq 0.6$ (Fig. 7) and $\alpha \geq 0.2$ (Fig. 8). Likewise, variational quantum circuits are predicted to display similar robustness against noise[38]. Let us emphasize that in the rest of our paper we use $\alpha = 1$.

In Fig. 9a, b we show the absolute value of the kernel obtained from simulation, and IBM hardware respectively. The difference between the two kernels can be attributed to hardware noise. To better show the role of
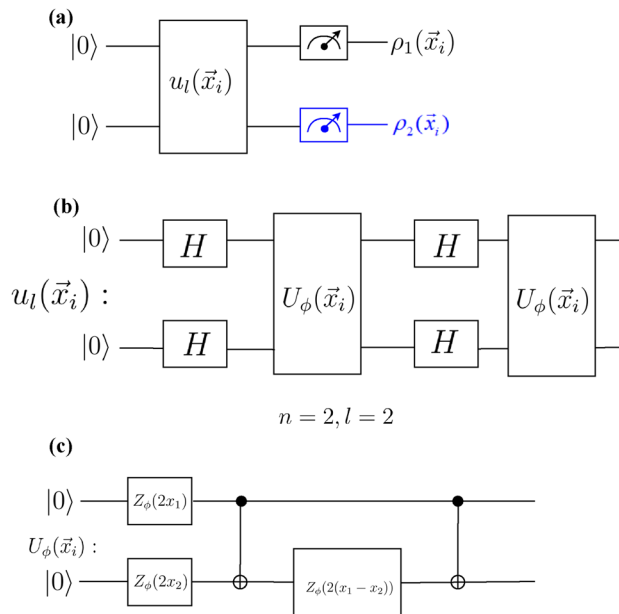

Figure 6. (a) A schematic picture of a two-qubit version of the projected kernel circuit. For the projected quantum kernel all qubits are measured, and moreover, all inputs are pure states. Here, for a two-qubit version, measuring the first qubit reconstructs the subsystem $\rho_1(\mathbf{x}_i)$, and then measuring the second qubit (indicated in blue) reconstructs $\rho_2(\mathbf{x}_i)$. (b) The gate decomposition for the unitary matrix $u^l(\mathbf{x}_i)$, where l is the number of the iterations of this gate decomposition (length of the circuit) for two qubits $n = 2$, and two iterations $l = 2$. (c) The gate components of the unitary operator $U_\phi(\mathbf{x}_i)$ adapted from[6]. Here, H is the Hadamard gate, and $Z_\phi = \text{diag}(1, e^{i\phi(\mathbf{x}_i)})$ is a single qubit phase gate.
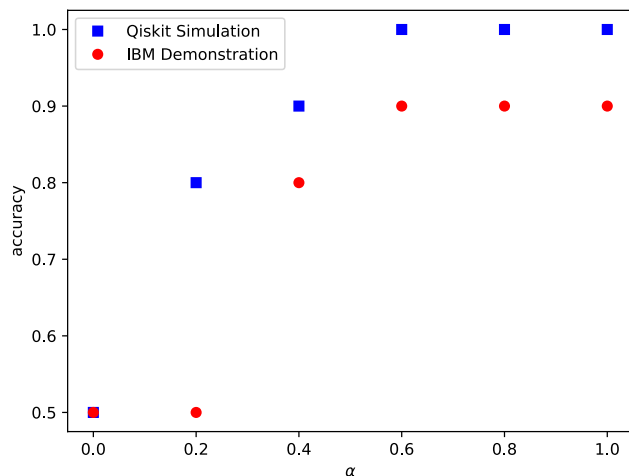
Figure 7. The accuracy as a function of the control qubit's purity for the same dataset as in Fig. 5 is shown. Note that when $\alpha = 0$, the state is in a completely mixed state, and when $\alpha = 1$, the state is pure. The blue curve indicates simulation results, and the red curve shows the results obtained from IBM hardware.
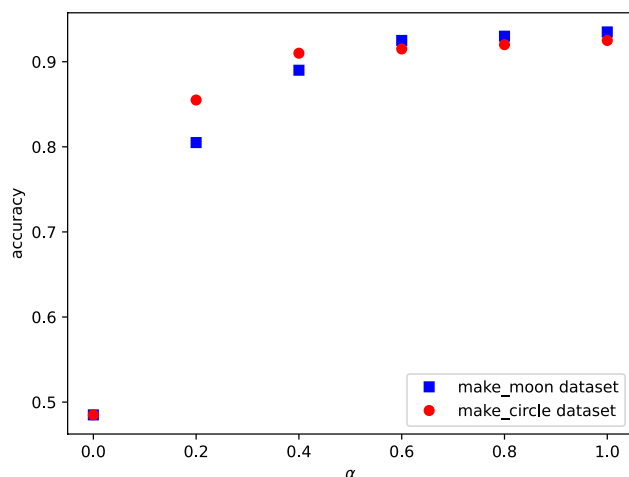

Figure 8. Simulation results of learning process for "make-moon" (blue curve), and "make-circle" dataset (red curve). The maximum accuracies, being 0.935, and 0.925, are both achieved at $\alpha = 1$.

noise in the kernel, we compare the diagonal elements of the kernel obtained from simulation and IBM hardware. As discussed earlier, in the ideal case $K(\mathbf{x}, \mathbf{x}) = 1$ (blue bar) but in practice one has $K(\mathbf{x}, \mathbf{x}) < 1$. In Fig. 9c, one finds a maximum difference of 0.610 between simulation and implementation, while the mean difference is 0.27.

Having access to the kernel, we can obtain the coherence consumption in our implementation from Eq. (19), as shown in Fig. 10. In accordance with Eq. (19), it can be seen from Fig. 10 that the coherence consumption is minimum (but not equal to zero in the IBM results) along the diagonal axes.

Our next step is to obtain the generated discord in our implementation based on Eq. (12). Equation (12) indicates that for estimating the discord, $\mathrm{tr}(U_n^2)$ must be estimated, which requires successive application of DQC1 evolution Eq. (2). Fig.11 shows the quantum discord. When comparing Fig.11 with Fig.10, it is also evident that the condition (13) is satisfied.

Discussion and conclusion

In[39], an upper bound on the generalization error of the fidelity quantum kernel model has been established, which is determined by the average purity of the encoded states. As highlighted in the same work, a noisier encoding process, when transferring data into quantum states, can lead to poorer training performance. Our empirical results, illustrated in Figs. 7 and 8 and based on three distinct datasets, support the idea that improving the purity of the control qubit (parameterized by α) enhances the accuracy of DQC1 kernel model. We believe that our DQC1 kernel framework provides a ground for rigorous theoretical studies for relating coherence consumption to generalization error.
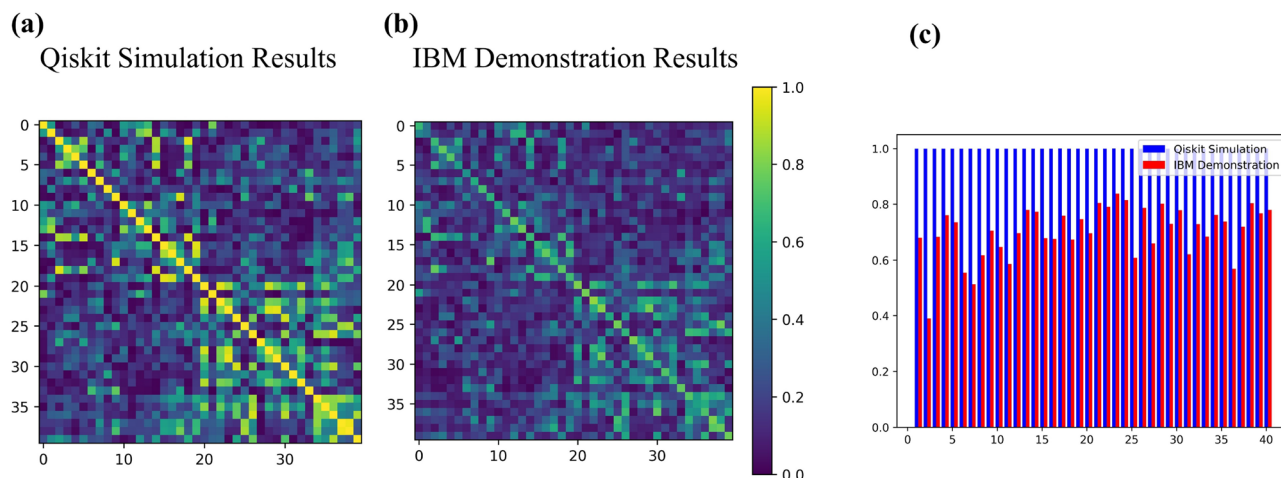
(a)
(b)
(c)

Figure 9. **(a)** The Qiskit simulation (left), and **(b)** IBM (right) results for the DQC1 quantum kernel with $n = 2$, and $l = 2$. IBM results have been obtained from "*ibm_perth*" device. **(c)** Diagonal elements of simulated (blue bars), and IBM (red bars) kernel matrices. The maximum difference between diagonal elements is 0.610, and the mean difference is 0.27.
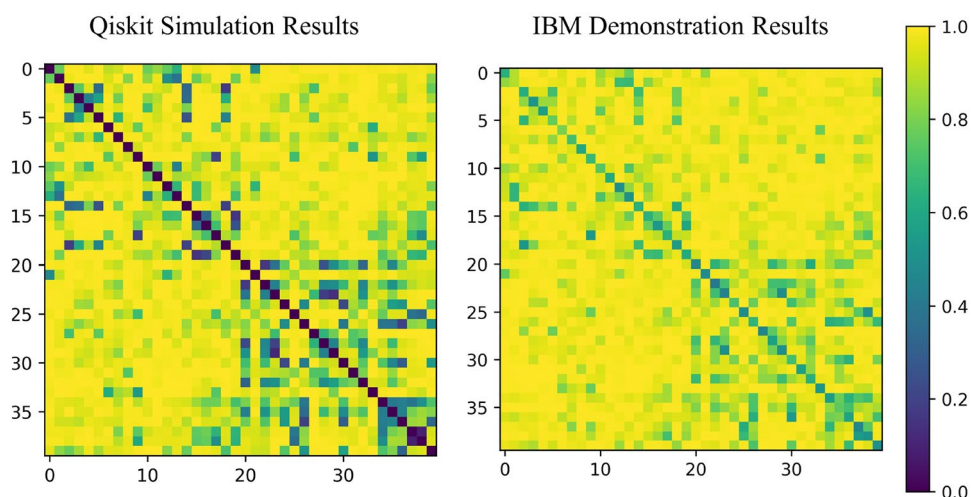

Figure 10. The Qiskit simulation results (left), and IBM results (right) for coherence consumption for the same dataset as in Fig. 5, and the circuit in Fig. 4.

We also comment on the connection between our work and the findings in[40], which investigate the exponential concentration of quantum kernels. Despite our method relying solely on measuring the control qubit, it is essential to emphasize that the number of required measurements scales exponentially with the number of target qubits, as the variance in the kernel exponentially approaches zero. Hence, we expect the same untrainability issues to hold for our kernel function.

In this study, we have investigated the application of the DQC1 model, a restricted computational model, to supervised machine learning tasks. Unlike the standard universal computational model, the DQC1 model relies on mixed states and does not incorporate quantum entanglement into the computation. We have presented a test of the DQC1 model's ability to solve supervised machine learning problems for some classically difficult kernels[6]. Despite requiring a greater number of gates than a similar protocol described in[6], since one needs to measure only the control qubit, our protocol still achieved a relatively high level of classification accuracy. Our proposal highlights the potential of utilizing quantum discord over entanglement in the presence of noise.

In a broader context, our work highlights the computational power of a single-qubit as a universal classifier[41,42]. It would be interesting to realize our protocol in the NMR setting[43]. We hope that this study will inspire further research on the integration of quantum coherence and quantum discord in machine learning.
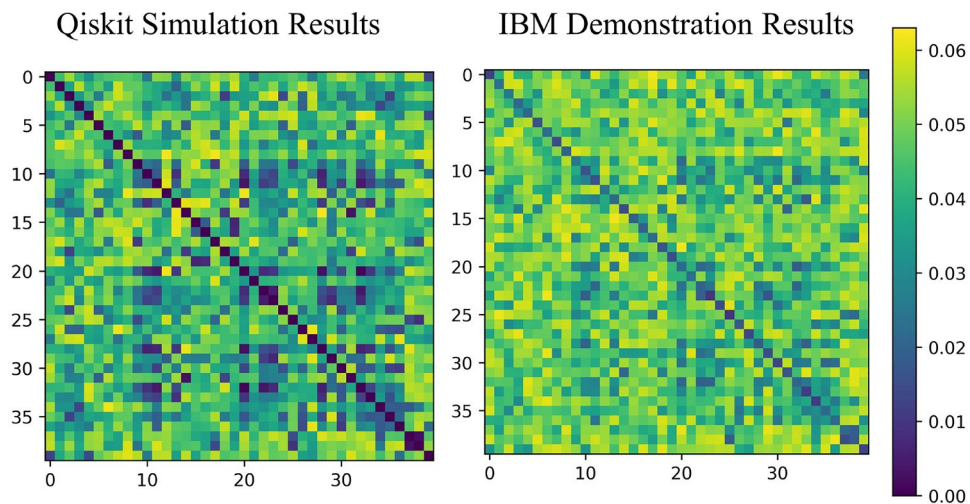
Figure 11. The Qiskit simulation results (left), and IBM results (right) for geometric discord for the same dataset as in Fig. 5, and the circuit in Fig. 4.

Data availability

The datasets and codes used for the generation of the plots of this manuscript are publicly available at https://github.com/mahsakarimii/The-power-of-one-clean-qubit-in-supervised-machine-learning.

References

1. Zhong, H.-S. *et al.* Quantum computational advantage using photons. *Science* **370**, 1460–1463 (2020).
2. Madsen, L. S. *et al.* Quantum computational advantage with a programmable photonic processor. *Nature* **606**, 75–81 (2022).
3. Preskill, J. Quantum computing in the Nisq era and beyond. *Quantum* **2**, 79 (2018).
4. Huang, H.-Y. *et al.* Quantum advantage in learning from experiments. *Science* **376**, 1182–1186 (2022).
5. Schuld, M., Bergholm, V., Gogolin, C., Izaac, J. & Killoran, N. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A* **99**, 032331 (2019).
6. Havlíček, V. *et al.* Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).
7. Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **113**, 130503 (2014).
8. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nat. Phys.* **10**, 631–633 (2014).
9. Gao, X., Anschuetz, E. R., Wang, S.-T., Cirac, J. I. & Lukin, M. D. Enhancing generative models via quantum correlations. *Phys. Rev. X* **12**, 021037 (2022).
10. Knill, E. & Laflamme, R. Power of one bit of quantum information. *Phys. Rev. Lett.* **81**, 5672 (1998).
11. Ollivier, H. & Zurek, W. H. Quantum discord: A measure of the quantumness of correlations. *Phys. Rev. Lett.* **88**, 017901 (2001).
12. Modi, K., Brodutch, A., Cable, H., Paterek, T. & Vedral, V. The classical-quantum boundary for correlations: Discord and related measures. *Reviews of Modern Physics* **84**, 1655 (2012).
13. Weedbrook, C., Pirandola, S., Thompson, J., Vedral, V. & Gu, M. How discord underlies the noise resilience of quantum illumination. *New J. Phys.* **18**, 043027 (2016).
14. Park, D. K., Rhee, J.-K.K. & Lee, S. Noise-tolerant parity learning with one quantum bit. *Phys. Rev. A* **97**, 032327 (2018).
15. Ghobadi, R., Oberoi, J. S. & Zahedinejhad, E. *The Power of One Qubit in Machine Learning*. arXiv preprint arXiv:1905.01390 (2019).
16. Vedaie, S. S., Noori, M., Oberoi, J. S., Sanders, B. C. & Zahedinejad, E. *Quantum Multiple Kernel Learning*. arXiv preprint arXiv:2011.09694 (2020).
17. Passante, G., Moussa, O., Trottier, D. & Laflamme, R. Experimental detection of nonclassical correlations in mixed-state quantum computation. *Phys. Rev. A* **84**, 044302 (2011).
18. Lanyon, B. P., Barbieri, M., Almeida, M. P. & White, A. G. Experimental quantum computing without entanglement. *Phys. Rev. Lett.* **101**, 200501 (2008).
19. Hor-Meyll, M. *et al.* Deterministic quantum computation with one photonic qubit. *Phys. Rev. A* **92**, 012337 (2015).
20. Wang, W. *et al.* Witnessing quantum resource conversion within deterministic quantum computation using one pure superconducting qubit. *Phys. Rev. Lett.* **123**, 220501 (2019).
21. Avron, H. & Toledo, S. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM (JACM)* **58**, 1–34 (2011).
22. Aaronson, S., Bouland, A., Kuperberg, G. & Mehraban, S. *The Computational Complexity of Ball Permutations*. 317–327 (2017).
23. Fujii, K. *et al.* Impossibility of classically simulating one-clean-qubit model with multiplicative error. *Phys. Rev. Lett.* **120**, 200502 (2018).
24. Poulin, D., Blume-Kohout, R., Laflamme, R. & Ollivier, H. Exponential speedup with a single bit of quantum information: Measuring the average fidelity decay. *Phys. Rev. Lett.* **92**, 177906 (2004).
25. Datta, A., Flammia, S. T. & Caves, C. M. Entanglement and the power of one qubit. *Phys. Rev. A* **72**, 042316 (2005).
26. Datta, A., Shaji, A. & Caves, C. M. Quantum discord and the power of one qubit. *Phys. Rev. Lett.* **100**, 050502 (2008).
27. Ma, J., Yadin, B., Girolami, D., Vedral, V. & Gu, M. Converting coherence to quantum correlations. *Phys. Rev. Lett.* **116**, 160407 (2016).
28. Baumgratz, T., Cramer, M. & Plenio, M. B. Quantifying coherence. *Phys. Rev. Lett.* **113**, 140401 (2014).

29. Passante, G., Moussa, O. & Laflamme, R. Measuring geometric quantum discord using one bit of quantum information. *Phys. Rev. A* **85**, 032325 (2012).
30. Dakić, B., Vedral, V. & Brukner, Č. Necessary and sufficient condition for nonzero quantum discord. *Phys. Rev. Lett.* **105**, 190502 (2010).
31. Tong, S. & Koller, D. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.* **2**, 45–66 (2001).
32. Hofmann, T., Schölkopf, B. & Smola, A. J. Kernel methods in machine learning. *Ann. Stat.* **36**, 1171–1220 (2008).
33. Schuld, M. & Killoran, N. Quantum machine learning in feature Hilbert spaces. *Phys. Rev. Lett.* **122**, 040504 (2019).
34. Karimi, M. *The Power of One Clean Qubit in Supervised Machine Learning. Codes are Available via this Link* (2022).
35. Madden, L. & Simonetto, A. Best approximate quantum compiling problems. *ACM Trans. Quantum Comput.* **3**, 1–29 (2022).
36. Huang, H.-Y. *et al.* Power of data in quantum machine learning. *Nat. Commun.* https://doi.org/10.1038/s41467-021-22539-9 *(2021)*.
37. Arute, F. *et al.* Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
38. Liu, J., Lin, Z. & Jiang, L. *Laziness, Barren Plateau, and Noise in Machine Learning*. arXiv preprint arXiv:2206.09313 (2022).
39. Heyraud, V., Li, Z., Denis, Z., Le Boité, A. & Ciuti, C. Noisy quantum kernel machines. *Phys. Rev. A* **106**, 052421 (2022).
40. Thanasilp, S., Wang, S., Cerezo, M. & Holmes, Z. *Exponential Concentration and Untrainability in Quantum Kernel Methods*. arXiv preprint arXiv:2208.11060 (2022).
41. Pérez-Salinas, A., Cervera-Lierta, A., Gil-Fuster, E. & Latorre, J. I. Data re-uploading for a universal quantum classifier. *Quantum* **4**, 226 (2020).
42. Dutta, T., Pérez-Salinas, A., Cheng, J. P. S., Latorre, J. I. & Mukherjee, M. Single-qubit universal classifier implemented on an ion-trap quantum device. *Phys. Rev. A* **106**, 012411 (2022).
43. Kusumoto, T., Mitarai, K., Fujii, K., Kitagawa, M. & Negoro, M. Experimental quantum kernel trick with nuclear spins in a solid. *npj Quantum Inf.* **7**, 94 (2021).

Acknowledgements

Author contributions

M.K performed simulation analysis. A.J-A wrote the optimization code to run the code on IBM hardware. C.S provided detailed feedback on the manuscript. R.G supervised the entire project and wrote the manuscript with input from M.K, A.J-A and C.S.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-023-46497-y.

Correspondence and requests for materials should be addressed to M.K. or R.G.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.