Large-scale quantum machine learning

Tobias Haug,[1,*] Chris N. Self,[1] and M. S. Kim[1]

[1] *QOLS, Blackett Laboratory, Imperial College London SW7 2AZ, UK*

Quantum computers promise to enhance machine learning for practical applications. Quantum machine learning for real-world data has to handle extensive amounts of high-dimensional data. However, conventional methods for measuring quantum kernels are impractical for large datasets as they scale with the square of the dataset size. Here, we measure quantum kernels using randomized measurements to gain a quadratic speedup in computation time and quickly process large datasets. Further, we efficiently encode high-dimensional data into quantum computers with the number of features scaling linearly with the circuit depth. The encoding is characterized by the quantum Fisher information metric and is related to the radial basis function kernel. We demonstrate the advantages and speedups of our methods by classifying images with the IBM quantum computer. Our approach is exceptionally robust to noise via a complementary error mitigation scheme. Using currently available quantum computers, the MNIST database can be processed within 220 hours instead of 10 years which opens up industrial applications of quantum machine learning.

Quantum machine learning aims to use quantum computers to enhance the power of machine learning [1, 2]. One possible route to quantum advantage in machine learning is the use of quantum embedding kernels [3–5], where quantum computers are used to encode data in ways that are difficult for classical machine learning methods [6–8]. Noisy intermediate scale quantum computers [9, 10] are capable of solving tasks difficult for classical computers [11, 12] and have shown promise in running proof-of-principle quantum machine learning applications [13–20]. However, major bottlenecks still limit the use of quantum hardware for machine learning with practical applications. Firstly, the cost of computing quantum kernels with conventional measurement methods scales quadratically with the size of the training dataset [5]. This quadratic scaling is a severe restriction, as most industrial relevant machine learning tasks rely on large amounts of data. Additionally, the data has to be encoded into the quantum computer in an efficient manner and generate a useful quantum kernel. Various encodings have been proposed [21, 22], however the number of features can be limited by the number of qubits [17, 18] and often the quantum kernel is characterized only in a heuristic manner. Finally, the inherent noise of quantum computers limits the quality of the experimental results. Error mitigation has been proposed to reduce the effect noise [23], however in general this requires a large amount of additional quantum computing resources [24].

Here, we use randomized measurements to calculate quantum kernels. The quantum computing time scales linearly with the size of the dataset for a rapid processing of large datasets. Additionally, we can reuse the collected measurement data to effectively mitigate the noise of quantum computers. To efficiently load high-dimensional data into the quantum computer, we demonstrate an encoding that scales linearly with the depth of parameterized quantum circuits (PQCs). The resulting quantum kernel is characterized with the quantum Fisher information metric (QFIM) and can be approximately described by the radial basis function kernel. We introduce the natural PQC (NPQC) with an exactly known QFIM and demonstrate its usefulness for quantum machine learning. We implement our approach on the IBM quantum computer to classify handwritten images of digits. Using this strategy, we estimate that large-scale datasets containing ten thousands of entries can be quickly processed for real-world applications of quantum machine learning.

Support vector machine— Our goal is to classify unlabeled test data by learning from labeled training data as shown in Fig.1a [2]. The dataset for the supervised learning task contains in total L entries. The i-th data item is described by a M-dimensional feature vector $\boldsymbol{x}_i$ with label y_i, which belongs to one of C possible classes. To learn and classify data, we use a kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ that is a measure of distance between feature vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. The kernel corresponds to an embedding of the M-dimensional data into a higher-dimensional space, where analysis of the data becomes easier [25]. In quantum kernel learning, we embed the data into the high-dimensional Hilbert space of the quantum computer and use it to calculate the kernel (see Fig.1b). With the kernels, we train a support vector machine (SVM) to find hyperplanes that separate two classes of data (see Fig.1c). The SVM is optimized using the kernels of the training dataset with a semidefinite program that can be efficiently solved with classical [26] or quantum computers [27, 28].

$$\max_{\boldsymbol{\alpha}} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{1}$$

subject to the conditions $\sum_i \alpha_i y_i = 0$ and $\alpha_i \geq 0$. After finding the optimal weights $\boldsymbol{\alpha}^*$, the SVM predicts the class of a feature vector $\boldsymbol{\eta}$ as $\text{sign}(\sum_i \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{\eta}) + b)$, where b is calculated from the weights. The SVM can be easily extended to C classes by solving C SVMs that separate each class from all other classes.

The power of the SVM highly depends on a good choice of kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, such that it must capture the essen-

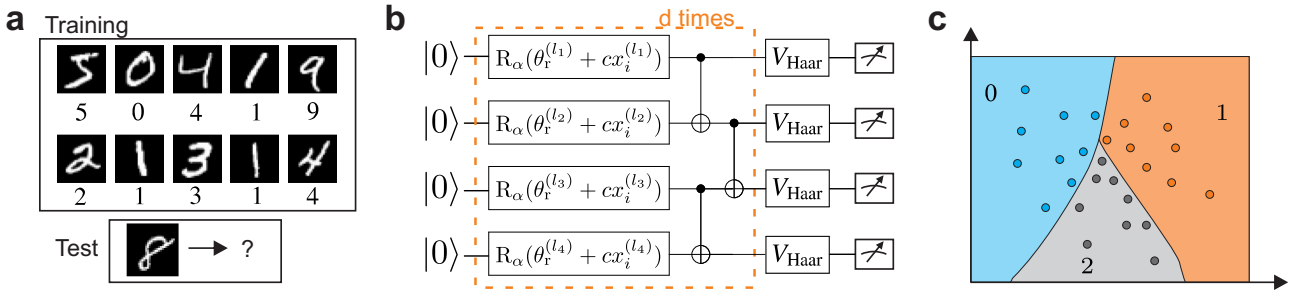

FIG. 1. **a)** Supervised learning to classify images of handwritten digits. By learning from a training set of labeled images, our goal is to identify previously unseen test data correctly. The support vector machine (SVM) learns using a kernel (Eq. (3)) which is a measure of distance between the data. **b)** The pixels of each image are converted to M-dimensional feature vectors $\boldsymbol{x}_i$ that are encoded as parameters $\boldsymbol{\theta}_i$ on a parameterized quantum circuit (PQC). We use hardware efficient PQCs of N qubits constructed from d layers of parameterized single qubit rotations and two-qubit entangling gates that can be efficiently implemented noisy quantum computers. The n-th index of the feature vector $x_i^{(n)}$ is mapped to a parameter of the single-qubit rotations via $\theta_i^{(n)} = \theta_{\text{r}}^{(n)} + cx_i^{(n)}$ (Eq. (2)), where $\boldsymbol{\theta}_{\text{r}}$ is a fixed reference parameter. The number of encoded features scales linearly with N and d. The kernel (Eq. (3)) is characterized by the quantum Fisher information metric (QFIM) $\mathcal{F}(\boldsymbol{\theta}_{\text{r}})$ and can be approximately described by the radial basis function kernel (Eq. (5)). We calculate the quantum kernel by measuring the PQC in randomized local bases of Haar random unitaries V_{Haar}. The quantum computation time scales linearly with the size L of the dataset. **c)** The SVM trained with the quantum kernel draws the decision boundaries (here shown for a two-dimensional feature vector space and three possible digits) that classify each feature vector to its corresponding label.

tial features of the dataset. Finding good kernels is a major challenge that has been widely studied in classical machine learning. In the following, we propose a powerful class of quantum kernels that can be implemented with currently available quantum computers. Then, we show how to compute kernels for large datasets and mitigate the noise inherent in real quantum devices.

Encoding— A crucial question is how to efficiently encode a feature vector with large M into a quantum computer while providing a useful kernel for machine learning. As shown in Fig.1b, we encode the M-dimensional feature vector $\boldsymbol{x}_i$ as M-dimensional parameter $\boldsymbol{\theta}_i$ of a PQC via

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_{\text{r}} + c\boldsymbol{x}_i \,, \tag{2}$$

where c is a scaling constant and $\boldsymbol{\theta}_{\text{r}}$ the reference parameter. We encode the feature vector in hardware efficient PQCs with N qubits and d layers of unitaries [29]. Each layer is composed of a product of parameterized single qubit rotations $V_l(\boldsymbol{\theta}_i^{(l)})$ and non-parameterized two-qubit entangling gates W_l that generate the quantum state $|\psi(\boldsymbol{\theta}_i)\rangle = \prod_{l=1}^{d} W_l V_l(\boldsymbol{\theta}_i^{(l)})|0\rangle^{\otimes N}$.

Our choice of quantum kernel measures the distance between two encoding states as given by the fidelity between $\rho(\boldsymbol{\theta}_i)$ and $\rho(\boldsymbol{\theta}_j)$ [7, 22]

$$K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \text{Tr}(\rho(\boldsymbol{\theta}_i)\rho(\boldsymbol{\theta}_j)) \,, \tag{3}$$

which for pure states $\rho(\boldsymbol{\theta}_i) = |\psi(\boldsymbol{\theta}_i)\rangle\langle\psi(\boldsymbol{\theta}_i)|$ reduces to $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = |\langle\psi(\boldsymbol{\theta}_i)|\psi(\boldsymbol{\theta}_j)\rangle|^2$.

We can formalize the expressive power of our encoding with the QFIM $\mathcal{F}(\boldsymbol{\theta})$, which is a $M \times M$ dimensional positive-semidefinite matrix that provides information about the kernel in the proximity of $\boldsymbol{\theta}$ [30]. For a pure state $|\psi\rangle = |\psi(\boldsymbol{\theta})\rangle$ it is given by $\mathcal{F}_{ij}(\boldsymbol{\theta}) = 4[\langle\partial_i\psi|\partial_j\psi\rangle - \langle\partial_i\psi|\psi\rangle\langle\psi|\partial_j\psi\rangle]$, where $\partial_j|\psi\rangle$ is the gradient in respect to the j-th element of $\boldsymbol{\theta}$ [31]. In the limit $c \to 0$ of encoding Eq. (2), the kernel of a pure quantum state can be written as

$$K(\boldsymbol{\theta}_{\text{r}}, \boldsymbol{\theta}_{\text{r}} + c\boldsymbol{x}_i) = 1 - \frac{c^2}{4}\boldsymbol{x}_i^{\text{T}}\mathcal{F}(\boldsymbol{\theta}_{\text{r}})\boldsymbol{x}_i = 1 - \frac{c^2}{4}\sum_{k=1}^{M}\lambda_k g_k \,, \tag{4}$$

where λ_k is the k-th eigenvalue of the QFIM $\mathcal{F}(\boldsymbol{\theta}_{\text{r}})$ and $g_k = |\langle\boldsymbol{x}_i, \boldsymbol{\mu}_k\rangle|^2$ is the inner product of the feature vector $\boldsymbol{x}_i$ and the k-th eigenvector $\boldsymbol{\mu}_k$ of $\mathcal{F}(\boldsymbol{\theta}_{\text{r}})$. The rank R (the number of non-zero eigenvalues) of $\mathcal{F}(\boldsymbol{\theta}_{\text{r}})$ is an important measure of the properties of the PQC and the encoding [30]. The $M-R$ eigenvectors $\boldsymbol{\mu}_k$ with $\lambda_k = 0$ have no effect on the kernel with $K(\boldsymbol{\theta}, \boldsymbol{\theta} + c\boldsymbol{\mu}_k) = 1$. Thus, feature vectors $\boldsymbol{x}_n$ that lie in the space of eigenvectors with eigenvalue zero cannot be distinguished using the kernel as they have the same value $K(\boldsymbol{\theta}, \boldsymbol{\theta} + c\boldsymbol{x}_n) = 1$. Further, the size of the eigenvalues λ_k determines how strongly the kernel changes in direction $\boldsymbol{\mu}_k$ of the feature space. By appropriately designing the QFIM as the weight matrix of the kernel, generalizing from data could be greatly enhanced [22, 30, 32]. For example, the feature subspace with eigenvalue 0 could be engineered such that it coincides with data that belongs to a particular class. Conversely, features that strongly differ between different classes could be tailored to have large eigenvalues such that they can be easily distinguished. For a PQC with N qubits the rank is upper bounded by $R \leq 2^{N+1}-2$, which is the maximal number of features that can be reliably distinguished by the kernel [30].

It has been recently shown that the kernel of pure

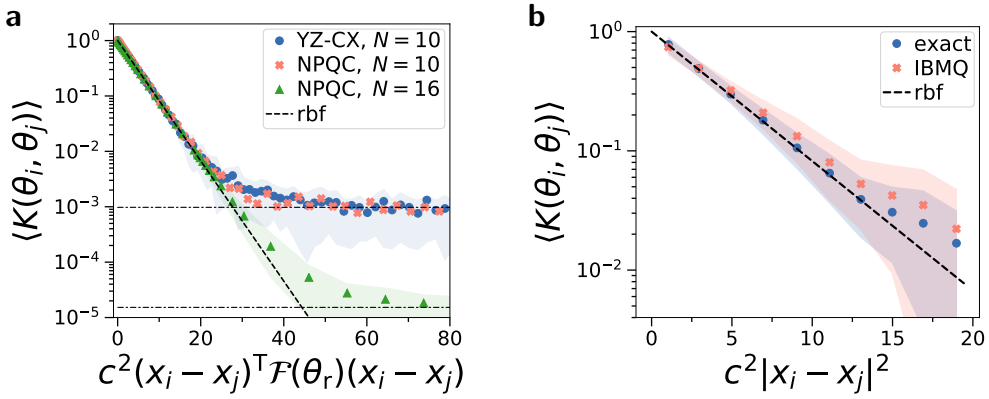

FIG. 2. **a)** Simulated kernel $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ as function of $(\boldsymbol{x}_i - \boldsymbol{x}_j)^{\mathrm{T}} \mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}})(\boldsymbol{x}_i - \boldsymbol{x}_j)$, which is the distance in feature space weighed with the QFIM $\mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}})$. The feature vectors $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ are randomly sampled and encoded with Eq. (2) into the PQC. We show two types of hardware efficient PQCs (see supplemental materials for construction). Shaded area is the standard deviation of the kernel. The quantum kernels are well approximated by radial basis function kernels (rbf, dashed line, Eq. (5)) until reaching very small values $K_{\min} = 2^{-N}$ (dash-dotted lines). PQCs have $N = 10$ qubits, $d = 10$ layers and we average over 50 random feature vectors. **b)** Experimental kernel $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ as function of distance between the feature vectors. We encode randomly chosen feature vectors in the NPQC with the QFIM $\mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}}) = I$. The quantum kernel generated by theory (blue dots) and via experimental results with IBM quantum computer (orange crosses) follows approximately the isotropic radial basis function kernel ($K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \exp(-\frac{c^2}{4}|\boldsymbol{x}_i - \boldsymbol{x}_j|^2)$, black line). Shaded area is standard deviation of the kernel. The NPQC has $N = 8$ qubits, $M = 36$ features and $d = 4$ layers. Experimental results from *ibmq_guadalupe* were performed with $r = 50$ randomized measurement settings, $s = 8192$ measurement samples and error mitigation with Eq. (7).

quantum states of hardware efficient PQCs can be approximated as Gaussian or radial-basis function kernels [33], which are one of the most popular non-linear kernels with wide application in various machine learning methods [34]. Specifically, for small enough c with the encoding Eq. (2), we can approximately describe the quantum kernel as

$$K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \approx \exp[-\frac{c^2}{4}(\boldsymbol{x}_i - \boldsymbol{x}_j)^{\mathrm{T}} \mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}})(\boldsymbol{x}_i - \boldsymbol{x}_j)]\,, \quad (5)$$

which is the radial basis function kernel with the QFIM as weight matrix $\mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}})$ [33]. While for general PQCs the QFIM is a priori not known, the NPQC has the special property that the QFIM takes a simple form with $\mathcal{F}(\boldsymbol{\theta}_{\mathrm{r}}) = I$, where I is the identity matrix and $\boldsymbol{\theta}_{\mathrm{r}}$ a particular reference parameter (see [35] and supplemental materials). The NPQC forms an approximate isotropic radial basis function kernel that can serve as a well characterised basis for quantum machine learning.

Measurement— We calculate the quantum kernels using randomized measurements [36, 37] by measuring quantum states in r randomly chosen single qubit bases. We prepare the quantum state $\rho(\boldsymbol{\theta}_i)$ and rotate into a random basis with the unitary $V^{(n)} = \otimes_{l=1}^{N} V_l^{(n)}$, where the single qubit unitaries $V_l^{(n)}$ are chosen according to the Haar measure over $\mathbb{SU}(2)$. Then, we measure s samples of the rotated state $V^{(n)}\rho(\boldsymbol{\theta}_i)V^{(n)\dagger}$ in the computational basis and estimate the probability $P_i^{(n)}(v_k)$ of measuring the computational basis state v_k for parameter $\boldsymbol{\theta}_i$ and basis n. This procedure is repeated for r different measurement bases and all L quantum states. The kernel $K_{\mathrm{b}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \mathrm{Tr}(\rho(\boldsymbol{\theta}_i)\rho(\boldsymbol{\theta}_j))$ via randomized measurements is then calculated as

$$K_{\mathrm{b}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \sum_{k,q} (-2)^{-D(k,q)} \sum_{n=1}^{r} P_i^{(n)}(k) P_j^{(n)}(q)\,, \quad (6)$$

where $D(k, q)$ is the Hamming distance that counts the number of bits that differ between the computational states v_k and v_q.

The statistical error of estimating the kernel $K_{\mathrm{b}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ in this way scales as $\propto 1/\sqrt{r}$ with the number of measurement bases r. The number s of measurement samples per basis scales exponentially with the number N of qubits [36, 37]. However, by using importance sampling, the number of measurements can be drastically reduced [38]. The number of measurements needed to determine all entries of the kernel matrix scales linearly with the dataset size L, allowing us to quickly process kernels of large datasets. Other commonly used measurement strategies such as the swap test [39, 40] or the inversion test [18] have to explicitly prepare both states $\rho(\boldsymbol{\theta}_i)$ and $\rho(\boldsymbol{\theta}_j)$ on the quantum computer and thus scale unfavorably with the square L^2 of the dataset size (see supplemental materials).

Error mitigation— In general, quantum computers are affected by noise, which will turn the prepared pure quantum state into a mixed state and may negatively affect the capability to learn. For depolarizing noise, we can use the information gathered in the process to mitigate its effect and infer the noiseless value of the kernel.

For global depolarizing noise, with a probability p_i the pure quantum state $|\psi(\boldsymbol{\theta}_i)\rangle$ is replaced with the completely mixed state $\rho_\mathrm{m} = I/2^N$, where I is the identity matrix. The resulting quantum state is the density matrix $\rho(\boldsymbol{\theta}_i) = (1-p_i)|\psi(\boldsymbol{\theta}_i)\rangle\langle\psi(\boldsymbol{\theta}_i)| + p_i\rho_\mathrm{m}$. The purity can be determined from the randomized measurements $\mathrm{Tr}(\rho(\boldsymbol{\theta}_i)^2) = K_\mathrm{b}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_i) = (1-p_i)^2 + \frac{p_i}{2^N}$ by reusing the same data used to compute the kernel entries. Using these purities, the depolarization probability p_i can be calculated by solving a quadratic equation [41]. With p_i and the kernel $K_\mathrm{b}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ affected by depolarizing noise, the mitigated kernel is given by

$$K_\mathrm{m}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \frac{K_\mathrm{b}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) - 2^{-N}}{(1-p_i)(1-p_j)} + \frac{1}{2^N}\,, \tag{7}$$

which simplifies for small p_i, p_j to $K_\mathrm{m}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \approx K_\mathrm{b}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)/\sqrt{\mathrm{Tr}(\rho(\boldsymbol{\theta}_i)^2)\mathrm{Tr}(\rho(\boldsymbol{\theta}_j)^2)}$.

Results— We now proceed to numerically and experimentally demonstrate our methods. First, we investigate the kernel of our encoding. In Fig.2a we simulate hardware efficient PQCs (YZ-CX and NPQC as defined in the supplemental materials) using qutip [42] and show that the quantum kernel is well described by a radial basis function kernel (Eq. (5), dashed line). The kernel diverges from the radial basis function kernel for small values of the kernel and reaches a plateau at $K_\mathrm{min} = \frac{1}{2^N}$, which is the fidelity of Haar random states [43]. In Fig.2b, we experimentally measure the kernel with an IBM quantum computer (*ibmq_guadalupe* [44]) and the NPQC using randomized measurements with error mitigation (Eq. (7)) and find that the mean value of the kernel matches well with the isotropic radial basis function kernel. See supplementary materials for details on the experiment and results regarding the YZ-CX PQC.

Next we address the statistical error introduced by estimating the kernel using randomised measurements and depolarizing noise p. In Fig.3a we simulate the average error

$$\Delta K = \frac{2}{L(L-1)} \sum_{i=1}^{L} \sum_{j=i+1}^{L} |K_\mathrm{m}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) - K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)| \tag{8}$$

of measuring the kernel $K_\mathrm{m}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ using randomized measurements with respect to its exact value $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ as function of number of measurement samples s. We find that there is a threshold of samples where the error becomes minimal. We are able to mitigate depolarizing noise to a noise-free level even for high p. In Fig.3b, we show the minimal number of samples s_min required to measure the kernel with an average error of at most $\Delta K < 0.1$ as function of depolarization noise p. The randomized measurement scheme works well even with substantial noise p, where we find a power law $s_\mathrm{min} \propto (1-p)^{-2}$.

Finally, to assess the overall performance of our approach we learn to classify handwritten 2D images of digits ranging from 0 to 9. The dataset contains $L = 1797$ 2D images of 8×8 pixels, where each pixel has an integer value between 0 and 16 [45]. We map the image to $M = 64$ dimensional feature vectors. For the YZ-CX PQC, we use all $M = 64$ features, whereas for the NPQC we perform a principal component analysis to reduce it to $M = 36$ features. We calculate the kernel of the full dataset and use a randomly drawn part of it as training data for optimizing the SVM with Scikit-learn [46]. The accuracy of the SVM is defined as the percentage of correctly classified test data, which are $L_\mathrm{test} = 200$ images that have not been used for training. The dataset is rescaled using the training data such that each feature has mean value zero and its variance is given by $\frac{1}{\sqrt{M}}$. We encode the feature vectors $\boldsymbol{x}_i$ via Eq. (2) with $c = 1$, where for the YZ-CX PQC we choose $\boldsymbol{\theta}_\mathrm{r}$ randomly and for the NPQC we define $\boldsymbol{\theta}_\mathrm{r}$ such that the QFIM is given by $\mathcal{F}(\boldsymbol{\theta}_\mathrm{r}) = I$ (see supplemental materials). We plot the accuracy of classifying test data with the SVM against the size of the training data in Fig.4a for the YZ-CX PQC and in Fig.4b for the NPQC. As kernels, we compare simulations with radial basis function kernel (rbf), the

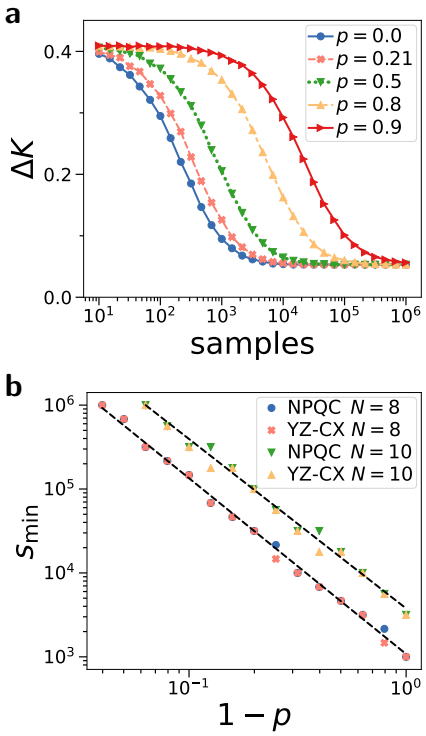

FIG. 3. **a)** Average error for measuring the kernel with randomized measurements ΔE as function of number of measurement samples s and the global depolarizing probability p. We use $r = 8$ measurement settings, $N = 8$ qubits and the YZ-CX PQC. **b)** Minimal number of measurement samples s_min needed to achieve an average error of at most $\Delta E < 0.1$ for varying depolarizing noise p. Dashed line is a power law $s_\mathrm{min} \propto (1-p)^{-2}$. Number of measurement settings is $r = 8$ for $N = 8$ and $r = 16$ for $N = 10$.

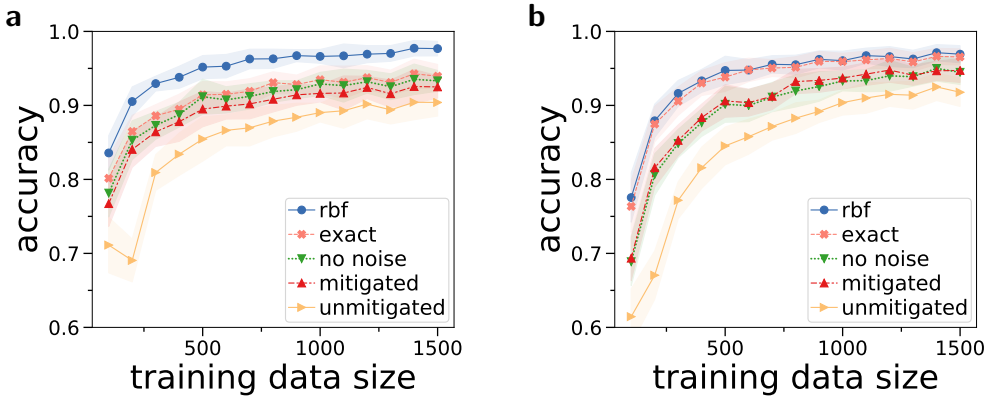

FIG. 4. Accuracy of classifying previously unseen handwritten digits correctly as function of the size of the training data. The shaded area is the standard deviation of the accuracy. We compare the isotropic radial basis function kernel (blue dots), exact quantum kernel (orange), noiseless simulation of randomized measurements (green), kernel of IBM quantum computer with randomized measurements using error mitigation (red, Eq. (7)) and no error mitigation (yellow). **a)** YZ-CX PQC with $M = 64$ features and **b)** NPQC with $M = 36$ features. Experiments were performed on *ibmq_guadalupe* using $s = 8192$ measurement samples, $N = 8$ qubits and $r = 8$ randomized measurement settings. The test data contains $L_{\text{test}} = 200$ entries with test and training data randomly drawn from the full dataset, this is repeated 20 times for each training data size.

exact simulated quantum kernel (exact) and a noiseless simulation of the randomized measurements (noiseless). For experimental data, we use an IBM quantum computer (*ibmq_guadalupe* [44], see supplementary materials for more details) to perform randomized measurements with error mitigation (mitigated) and without error mitigation (unmitigated). The accuracy improves steadily with increased number of training data for all kernels. Our error mitigation scheme (Eq. (7)) substantially improves the accuracy of the SVM trained with experimental data to nearly the level of the noiseless simulation of the randomized measurements. The randomized measurements have a lower accuracy compared to the exact quantum kernel as we use only a relatively small number r of randomized measurement settings. For the NPQC (Fig.4b) the exact quantum kernel shows nearly the same accuracy as the radial basis function kernel, whereas for the YZ-CX PQC the quantum kernel performs slightly worse, likely indicating the QFIM does not optimally capture the structure of the data. The depolarizing probability of the IBM quantum computer is estimated as $p \approx 0.36$ for the NPQC and $p \approx 0.39$ for the YZ-CX. To measure the kernel of the dataset, we require in total $sLr \approx 1.2 \cdot 10^8$ experiments, whereas with the inversion test the quantum computation time would be a factor of 100 longer with $sL(L-1)/2 \approx 1.3 \cdot 10^{10}$ experiments. In the supplemental materials, we show the accuracy of the training data and the confusion matrices.

Discussion— We demonstrated machine learning of large datasets with quantum computers using randomized measurements. The machine learning data is encoded into hardware efficient PQCs, where the number M of features scales linearly with the depth d of the circuit and number N of qubits. The kernel is characterized by the QFIM and its eigenvalues and eigenvectors [30]. As the behavior of the kernel is crucial for effectively learning and generalizing data, future work could design the QFIM to improve the capability of quantum machine learning models. We demonstrated the NPQC with a simple and exactly known QFIM, which could be a useful basis to study quantum machine learning on large quantum computers. The relation of our PQCs and radial basis function kernels [33] gives us a strong indication that our encoding is at least as powerful as classical machine learning kernels and could be used to study the power of quantum machine learning [47]. However, we stress that the description as radial basis function kernels is only approximate and fails for very small kernel values, which may hide possible quantum advantages [7].

We mitigate the noise occurring in the quantum computer by using data sampled during the measurements of the kernel. We find that the number s_{min} of measurement samples needed to mitigate depolarizing noise scales as $s_{\text{min}} \propto (1-p)^{-2}$, allowing us to extract kernels even from very noisy quantum computers. We successfully apply this model to mitigate the noise of the IBM quantum computer. While the noise model of quantum computers is known to be complex, the depolarizing noise model is sufficient to mitigate the noise of quantum kernels [41]. We note that noise induced errors can actually be beneficial to machine learning as the capability to generalize from data improves with increasing noise [32].

In general, the number of measurements needed for the randomized measurement scheme scales exponentially with the number N of qubits [36, 37]. However, various approaches can mitigate this. Importance sampling can drastically reduce the number of measurements needed [38]. In other settings adaptive measurements

have been proposed to improve the scaling of measurement costs [48], as well as other approaches such as shadow tomography [49]. The choice of an effective set of measurements could be included in the machine learning task as hyper-parameters to be optimised. To reduce the number of qubits, one could combine our approach with quantum autoencoders to transform the encoding quantum state into a subspace with less qubits that captures the essential information of the kernel [50]. Alternatively, one could trace out most of the qubits of a many-qubit quantum state $\rho(\boldsymbol{\theta}_i)$ such that a subystem A with a lower number of qubits remains. Then, randomized measurements can efficiently determine the kernel $\mathrm{Tr}(\rho^A(\boldsymbol{\theta}_i)\rho^A(\boldsymbol{\theta}_j))$. It would be worthwhile to investigate the learning power of kernels generated from subsystems of quantum states that possess quantum advantage [6, 7].

Our method scales linearly with dataset size L and provides a quadratic speedup compared to conventional measurement methods such as the inversion test, allowing us to compute large datasets in a reasonable time. The measurements for the kernel do not have to be performed on the same machine, but can be freely mixed and matched between different quantum computers or quantum simulators [51], which can further reduce computation time by distributing and parallelizing the quantum computation tasks. Our encoding can load up to $M = 2^{N+1}-2$ features onto N qubits such that the popular MNIST dataset for classifying 2D images of handwritten digits with 28×28 pixels could fit already on $N = 9$ qubits [52]. Current state of the art quantum computers measure about 5000 quantum states per second [11, 12]. Assuming $s = 8192$ measurement samples and $r = 8$ measurement settings, our method can process the full MNIST training dataset with $L_{\text{train}} = 60000$ entries in about 220 hours of quantum processing time. In contrast, the inversion or swap test would require at least 10 years with $s = 1000$ samples. With our scheme, the currently available quantum computers can be benchmarked with large-scale datasets against classical machine learning algorithms and explore industrially relevant applications of quantum machine learning.

Acknowledgements— We acknowledge discussions with Kiran Khosla and Alistair Smith. This work is supported by a Samsung GRC project and the UK Hub in Quantum Computing and Simulation, part of the UK National Quantum Technologies Programme with funding from UKRI EPSRC grant EP/T001062/1. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

* thaug@ic.ac.uk

[1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).
[2] M. Schuld and F. Petruccione, *Supervised learning with quantum computers*, Vol. 17 (Springer, 2018).
[3] M. Schuld and N. Killoran, Quantum machine learning in feature hilbert spaces, Physical review letters **122**, 040504 (2019).
[4] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Physical Review A **103**, 032430 (2021).
[5] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning, arXiv:2001.03622 (2020).
[6] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, arXiv preprint arXiv:2010.02174 (2020).
[7] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, Power of data in quantum machine learning, Nature communications **12**, 1 (2021).
[8] H.-Y. Huang, R. Kueng, G. Torlai, V. V. Albert, and J. Preskill, Provably efficient machine learning for quantum many-body problems, arXiv:2106.12627 (2021).
[9] J. Preskill, Quantum computing in the nisq era and beyond, Quantum **2**, 79 (2018).
[10] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum (nisq) algorithms, arXiv:2101.08448 (2021).
[11] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).
[12] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, *et al.*, Strong quantum computational advantage using a superconducting quantum processor, arXiv preprint arXiv:2106.14734 (2021).
[13] K. Bartkiewicz, C. Gneiting, A. Černoch, K. Jiráková, K. Lemr, and F. Nori, Experimental kernel-based quantum machine learning in finite feature space, Scientific Reports **10**, 1 (2020).
[14] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, Quantum classifier with tailored quantum kernel, npj Quantum Information **6**, 1 (2020).
[15] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, *et al.*, Quantum machine learning in high energy physics, Machine Learning: Science and Technology (2020).
[16] E. Peters, J. Caldeira, A. Ho, S. Leichenauer, M. Mohseni, H. Neven, P. Spentzouris, D. Strain, and G. N. Perdue, Machine learning of high dimensional data on a noisy quantum processor, arXiv preprint arXiv:2101.09581 (2021).
[17] S. L. Wu, S. Sun, W. Guan, C. Zhou, J. Chan, C. L. Cheng, T. Pham, Y. Qian, A. Z. Wang, R. Zhang, *et al.*, Application of quantum machine learning using the quantum kernel algorithm on high energy physics analysis at the lhc, arXiv:2104.05059 (2021).
[18] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Super-

vised learning with quantum-enhanced feature spaces, Nature **567**, 209 (2019).
[19] S. Johri, S. Debnath, A. Mocherla, A. Singh, A. Prakash, J. Kim, and I. Kerenidis, Nearest centroid classification on a trapped ion quantum computer, arXiv preprint arXiv:2012.04145 (2020).
[20] H.-L. Huang, Y. Du, M. Gong, Y. Zhao, Y. Wu, C. Wang, S. Li, F. Liang, J. Lin, Y. Xu, *et al.*, Experimental quantum generative adversarial networks for image generation, arXiv:2010.06201 (2020).
[21] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, Quantum **4**, 226 (2020).
[22] M. Schuld, Quantum machine learning models are kernel methods, arXiv:2101.11020 (2021).
[23] K. Temme, S. Bravyi, and J. M. Gambetta, Error mitigation for short-depth quantum circuits, Physical review letters **119**, 180509 (2017).
[24] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, Hybrid quantum-classical algorithms and quantum error mitigation, Journal of the Physical Society of Japan **90**, 032001 (2021).
[25] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond* (Adaptive Computation and Machine Learning Series, 2018).
[26] H. Wolkowicz, R. Saigal, and L. Vandenberghe, *Handbook of semidefinite programming: theory, algorithms, and applications*, Vol. 27 (Springer Science & Business Media, 2012).
[27] F. G. Brandão, A. Kalev, T. Li, C. Y.-Y. Lin, K. M. Svore, and X. Wu, Quantum sdp solvers: Large speedups, optimality, and applications to quantum learning, arXiv:1710.02581 (2017).
[28] K. Bharti, T. Haug, V. Vedral, and L.-C. Kwek, Nisq algorithm for semidefinite programming, arXiv:2106.03891 (2021).
[29] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).
[30] T. Haug, K. Bharti, and M. S. Kim, Capacity and quantum geometry of parametrized quantum circuits, arXiv:2102.01659 (2021).
[31] J. J. Meyer, Fisher information in noisy intermediate-scale quantum applications, arXiv:2103.15191 (2021).
[32] L. Banchi, J. Pereira, and S. Pirandola, Generalization in quantum machine learning: a quantum information perspective, arXiv:2102.08991 (2021).
[33] T. Haug and M. S. Kim, Optimal training of variational quantum algorithms without barren plateaus, arXiv:2104.14543 (2021).
[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning* (MIT press, 2016).
[35] T. Haug and M. S. Kim, Natural parameterized quantum circuit, arXiv:2107.14063 (2021).
[36] A. Elben, B. Vermersch, C. F. Roos, and P. Zoller, Statistical correlations between locally randomized measurements: A toolbox for probing entanglement in many-body quantum states, Physical Review A **99**, 052323 (2019).
[37] A. Elben, B. Vermersch, R. van Bijnen, C. Kokail, T. Brydges, C. Maier, M. K. Joshi, R. Blatt, C. F. Roos, and P. Zoller, Cross-platform verification of intermediate scale quantum devices, Physical review letters **124**, 010504 (2020).
[38] A. Rath, R. van Bijnen, A. Elben, P. Zoller, and B. Vermersch, Importance sampling of randomized measurements for probing entanglement, arXiv:2102.13524 (2021).
[39] H. Buhrman, R. Cleve, J. Watrous, and R. De Wolf, Quantum fingerprinting, Physical Review Letters **87**, 167902 (2001).
[40] C.-H. Nguyen, K.-W. Tseng, G. Maslennikov, H. Gan, and D. Matsukevich, Experimental swap test of infinite dimensional quantum states, arXiv preprint arXiv:2103.10219 (2021).
[41] J. Vovrosh, K. E. Khosla, S. Greenaway, C. Self, M. Kim, and J. Knolle, Efficient mitigation of depolarizing errors in quantum simulations, arXiv e-prints , arXiv (2021).
[42] J. R. Johansson, P. D. Nation, and F. Nori, Qutip: An open-source python framework for the dynamics of open quantum systems, Computer Physics Communications **183**, 1760 (2012).
[43] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nature communications **9**, 4812 (2018).
[44] *ibmq_guadalupe* (v1.3.4) IBM Quantum team. Retrieved from https://quantum-computing.ibm.com (2021).
[45] C. Kaynak, Methods of combining multiple classifiers and their applications to handwritten digit recognition, Unpublished master's thesis, Bogazici University (1995).
[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research **12**, 2825 (2011).
[47] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, The power of quantum neural networks, Nature Computational Science **1**, 403 (2021).
[48] G. García-Pérez, M. A. Rossi, B. Sokolov, F. Tacchino, P. K. Barkoutsos, G. Mazzola, I. Tavernelli, and S. Maniscalco, Learning to measure: adaptive informationally complete povms for near-term quantum algorithms, arXiv:2104.00569 (2021).
[49] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nature Physics **16**, 1050 (2020).
[50] J. Romero, J. P. Olson, and A. Aspuru-Guzik, Quantum autoencoders for efficient compression of quantum data, Quantum Sci. Technol. **2**, 045001 (2017).
[51] D. Zhu, Z.-P. Cian, C. Noel, A. Risinger, D. Biswas, L. Egan, Y. Zhu, A. M. Green, A. Maksymov, Y. Nam, M. Cetina, N. M. Linke, M. Hafezi, and C. Monroe, Cross-platform comparison of arbitrary quantum computations, arXiv:2107.11387 (2021).
[52] L. Deng, The mnist database of handwritten digit images for machine learning research [best of the web], IEEE Signal Processing Magazine **29**, 141 (2012).
[53] H. Abraham *et al.*, Qiskit: An open-source framework for quantum computing (2019).
[54] S. Sivarajah, S. Dilkes, A. Cowtan, W. Simmons, A. Edgington, and R. Duncan, tket: a retargetable compiler for nisq devices, Quantum Science and Technology **6**, 014003 (2020).

Parameterized quantum circuits

We use two different types of PQCs. Both consist of d layers of unitaries $U_l(\boldsymbol{\theta}_l)$, generating the quantum state $U(\boldsymbol{\theta})|0\rangle = \prod_{l=1}^{d} U_l(\boldsymbol{\theta}_l)|0\rangle^{\otimes N}$ parameterized by the M-dimensional parameter vector $\boldsymbol{\theta} \in \mathbb{R}^M$.

In Fig.5a, we show the first circuit we use, which we call the NPQC. The first layer is $2N$ single qubit rotations around the y and z axis for each qubit n with $U_1 = \prod_{n=1}^{N} R_z^{(n)}(\theta_{1,z}^{(n)}) R_y^{(n)}(\theta_{1,y}^{(n)})$. Here, $R_\alpha^{(n)}(\theta) = \exp(-i\frac{\theta}{2}\sigma_n^\alpha)$, $\alpha \in \{x, y, z\}$ and σ_n^x, σ_n^y, σ_n^z are the Pauli matrices for qubit n. Each additional layer $l > 1$ is a product of two qubit entangling gates and N parameterized single qubit rotations defined as $U_l(a_l) = \prod_{k=1}^{N/2} [R_z^{(2k-1)}(\theta_{l,z}^{(2k-1)}) R_y^{(2k-1)}(\theta_{l,y}^{(2k-1)})] U_{\text{ent}}(a_l)$, where $U_{\text{ent}}(a_l) = \prod_{k=1}^{N/2} \text{CPHASE}(2k-1, 2k+2a_l) R_y^{(2k-1)}(\pi/2)$ and $\text{CPHASE}(n, m)$ is the controlled σ^z gate for qubit index n, m, where indices larger than N are taken modulo. The entangling layer $U_l(0)$ is shown as example in Fig.5b. The shift factor $a_l \in \{0, 1, \ldots, N/2-1\}$ for layer l is given by the recursive rule shown in the following. Initialise a set $A = \{0, 1, \ldots, N/2-1\}$ and $s = 1$. In each iteration, pick and remove one element r from A. Then set $a_s = r$ and $a_{s+q} = a_q$ for $q = \{1, \ldots, s-1\}$. As the last step, we set $s = 2s$. We repeat this procedure until no elements are left in A or a target depth d is reached. One can have maximally $d_{\max} = 2^{N/2}$ layers with in total $M = N(d+1)$ parameters. The NPQC has a QFIM $\mathcal{F}(\boldsymbol{\theta}_{\text{r}}) = I$, I being the identity matrix, for the reference parameter $\boldsymbol{\theta}_{\text{r}}$ given by

$$\mathcal{F}(\boldsymbol{\theta}_{\text{r}}) = I \;\text{ for }\; \theta_{\text{r},l,y}^{(n)} = \pi/2\,, \;\; \theta_{\text{r},l,z}^{(n)} = 0\,. \tag{9}$$

Close to this reference parameter, the QFIM remains approximately close to being an identity matrix. When implementing the NPQC for the IBM quantum computer, we choose the sift factor a_l such that only nearest-neighbor CPHASE gates arranged in a chain appear.

The second type of PQC used is shown in Fig.5c, which we call YZ-CX. It consists of d layers of parameterized single qubit y and z rotations, followed by CNOT gates. The CNOT gates arranged in a one-dimensional chain, acting on neighboring qubits. Every even layer l, the CNOT gates are shifted by one qubit. Redundant single qubit rotations that are left over at the edges of the chain are removed.

Measurement of kernels

In Fig.6, we explain the different methods to measure kernels of L quantum states. In this paper, we use the randomized measurements method shown in Fig.6a. The number of required measurements to measure all possible pairs of kernels scales linearly with dataset size L. The inversion test is shown in Fig.6b and swap test in Fig.6c. The number of required measurements scale with the square L^2 of the dataset size.

Experimental kernel of YZ-CX PQC

In Fig.7, we show experimental data of the kernel for the YZ-CX PQC using *ibmq_guadalupe*.

IBM Quantum implementation details

Our PQC circuits are constructed as parameterised circuits with Qiskit [53]. These parameterised circuits are first transpiled then bound for each data point and randomised measurement unitary, ensuring that all circuits submitted have the same structure and use the same set of device qubits. Transpiling is handled by the pytket python package [54] using *rebase*, *placement* and *routing* passes with no additional optimisations (IBMQ default passes with optimisation level 0).

The IBMQ results presented in Fig. 2 and Fig. 4 were collected between 22nd July 2021 and 30th July 2021 using the *ibmq_guadalupe* chip [44]. Fig. 2 required the execution of $100 \times 50 = 5000$ circuits and Fig. 4 involved $1790 \times 8 = 14,320$ circuits, each with 8192 measurement shots. For comparison, applying the inversion test to the same handwritten digit dataset used for Fig. 4 would have required the execution of $1790 \times 1790 \approx 3.2 \times 10^6$ circuits. Circuits were executed on IBM quantum devices using the circuit queue API. Job submissions were batched in such a way that all measurement circuits for a data point were submitted and executed together.

Beyond the error mitigation procedure described in the main text we carry out no further error mitigation, specifically we do not apply measurement error mitigation.

Training accuracy

In Fig.8, we plot the accuracy of classifying the training data with the SVM for the YZ-CX PQC and NPQC. The accuracy is defined as the percentage of training data that is correctly identified.

Confusion matrix

We now show the confusion matrices for the test data. The confusion matrix shows what label is predicted by the SVM in respect to its true label of the test data. The diagonal are the correctly classified digits, whereas the off-diagonals show the number of times a digit was miss-classified. In Fig.9, we show the confusion matrix

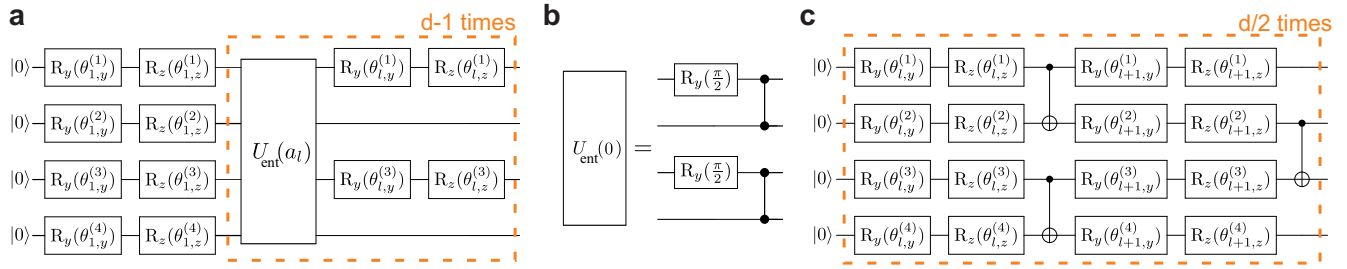

FIG. 5. **a)** The NPQC for N qubits and d layers, which is a hardware efficient PQC composed of single qubit rotations and CPHASE gates. For the reference parameter $\boldsymbol{\theta}_\mathrm{r}$, the QFIM is the identity matrix. **b)** Example of the entangling layer for the NPQC, which is composed of $N/2$ non-overlapping CPHASE gates and y rotations by $pi/2$. **c)** YZ-CX PQC, which is a hardware efficient circuit consisting of single qubit rotations and CNOT gates arranged in an alternating fashion in even and odd layers l.

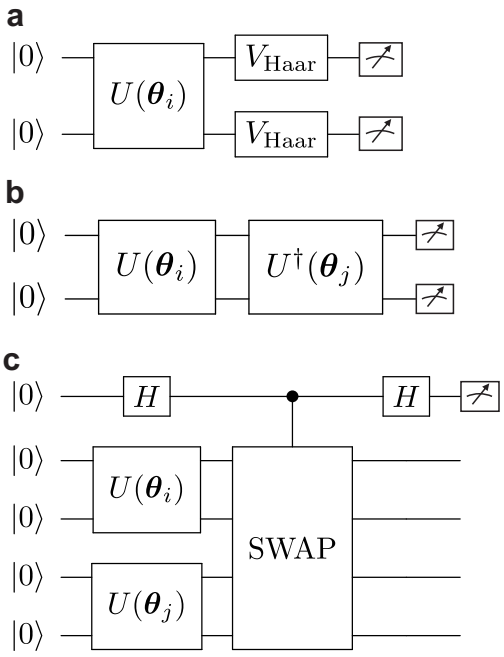

FIG. 6. Quantum circuits to measure kernel $K(\boldsymbol{\theta}_i), \boldsymbol{\theta}_j)$ of L quantum states $U(\boldsymbol{\theta}_i)|0\rangle$. **a)** Randomized measurement scheme. Prepare state $|\psi(\boldsymbol{\theta}_i\rangle$, rotate into randomized basis given by single qubit Haar random unitaries V_Haar and measure in computational basis. By post-processing sampled states one can extract kernel. Number of measurements scales linearly with L. **b)** Inversion test. Prepare $U^d agger(\boldsymbol{\theta}_j)U(\boldsymbol{\theta}_i)|0\rangle$ and measure probability of zero state $|0\rangle$. Scales with L^2. **c)** Swap test. Prepare $U(\boldsymbol{\theta}_j) \otimes U(\boldsymbol{\theta}_i)|0\rangle$ on twice the number of qubits and perform controlled SWAP gate with an ancilla. Kernel is determined by measuring ancilla only. Scales as L^2.

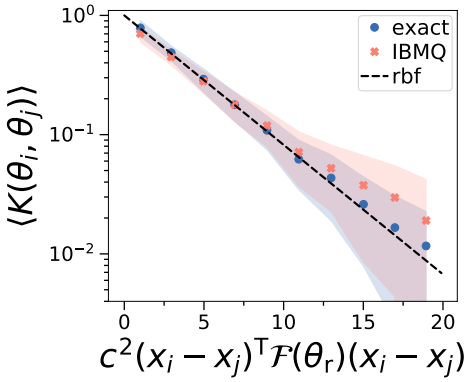

FIG. 7. Experimental kernel $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ as function of distance between the feature vectors $\boldsymbol{x}_i$, $\boldsymbol{x}_j$. We encode randomly chosen feature vectors via $\boldsymbol{\theta}_i = \boldsymbol{\theta}_\mathrm{r} + c\boldsymbol{x}_i$ in the YZ-CX PQC, where $\boldsymbol{\theta}_\mathrm{r} \in [0, 2\pi]$ is a randomly chosen reference parameter. The QFIM $\mathcal{F}(\boldsymbol{\theta}_\mathrm{r})$ is calculated numerically. The quantum kernel generated by theory (blue dots) and via experimental results with IBM quantum computer (orange crosses) follows approximately a weighted radial basis function kernel $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \exp[-\frac{c^2}{4}(\boldsymbol{x}_i - \boldsymbol{x}_j)^\mathrm{T}\mathcal{F}(\boldsymbol{\theta}_\mathrm{r})(\boldsymbol{x}_i - \boldsymbol{x}_j)]$ (black line). Shaded area is standard deviation of the kernel. The YZ-CX PQC has $N = 8$ qubits, $M = 72$ features and $d = 5$ layers. Experimental results from *ibmq_guadalupe* were performed with $r = 50$ randomized measurement settings, $s = 8192$ measurement samples and error mitigation.

for the NPQC. and in Fig.10 we show the confusion matrix for the YZ-CX PQC. We find that the actual digit 8 is often predicted to be the digit 1. Then, likely confusions are that digit 3 is assumed to be 8 and digit 9 is assumed to be 8. We find these confusions consistently in all kernels. While for the NPQC, radial basis function kernel and quantum kernel give nearly the same confusion matrix, we find substantial differences for the YZ-CX PQC. The reason is that while NPQC is an approximate isotropic radial basis function kernel, the YZ-CX PQC is an approximate radial basis function kernel with a weight matrix given by the QFIM. This weight matrix seems to reduce the accuracy of the classification.

Product state as analytic radial basis function kernel

We show that product states form an exact radial basis function kernel. We use the following N qubit quantum

state

$$|\psi(\boldsymbol{\theta})\rangle = \bigotimes_{n=1}^{N}(\cos(\frac{\theta_n}{2})|0\rangle + \sin(\frac{\theta_n}{2})|1\rangle)\,. \tag{10}$$

The QFIM is given by $\mathcal{F}(\boldsymbol{\theta}) = I$, where I is the M-dimensional identity matrix and $\mathcal{F}_{ij} = 4[\langle\partial_i\psi|\partial_j\psi\rangle - \langle\partial_i\psi|\psi\rangle\langle\psi|\partial_j\psi\rangle]$. The kernel of two states parameterized by $\boldsymbol{\theta}$, $\boldsymbol{\theta}'$ is given by

$$K(\boldsymbol{\theta},\boldsymbol{\theta}') = \left|\langle\psi(\boldsymbol{\theta})|\psi(\boldsymbol{\theta}')\rangle\right|^2 = \prod_{n=1}^{N}(1+\frac{1}{2}\cos(\Delta\theta_n)) \tag{11}$$

where we define $\Delta\boldsymbol{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}'$ as the difference between the two parameter sets. We now assume $|\Delta\theta_n| \ll 1$ and that all the differences of the parameters are equal $\Delta\theta_1 = \cdots = \Delta\theta_N$. We then find in the limit of many qubits N

$$K(\boldsymbol{\theta},\boldsymbol{\theta}') \approx \prod_{n=1}^{N}(1-\frac{1}{4}\Delta\theta_n^2) \xrightarrow[N\to\infty]{} \exp(-\frac{1}{4}\sum_{n=1}^{N}\Delta\theta_n^2)\,, \tag{12}$$

which gives us the radial basis function kernel.

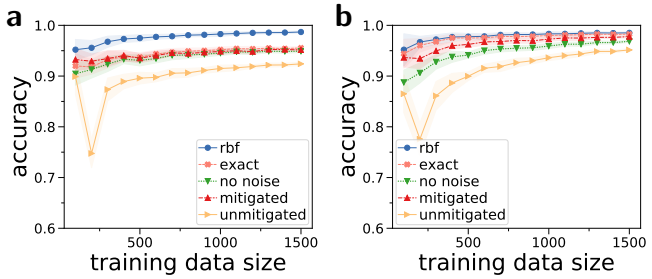

FIG. 8. Accuracy of classifying training data as function of the size of the training data. The shaded area is the standard deviation of the accuracy. We use same data as in the main text. We compare the radial basis function kernel (blue dots), exact quantum kernel (orange), noiseless simulation of randomized measurements (green), kernel of IBM quantum computer with randomized measurements using error mitigation (red) and no error mitigation (yellow). **a)** YZ-CX PQC with $M = 64$ features for and **b)** NPQC with $M = 36$ features. We use the *ibmq_guadalupe* with $s = 8192$ measurement samples, $N = 8$ qubits and $r = 8$ randomized measurement settings. The training data is randomly drawn from the full dataset, which is repeated 20 times for each training data size.

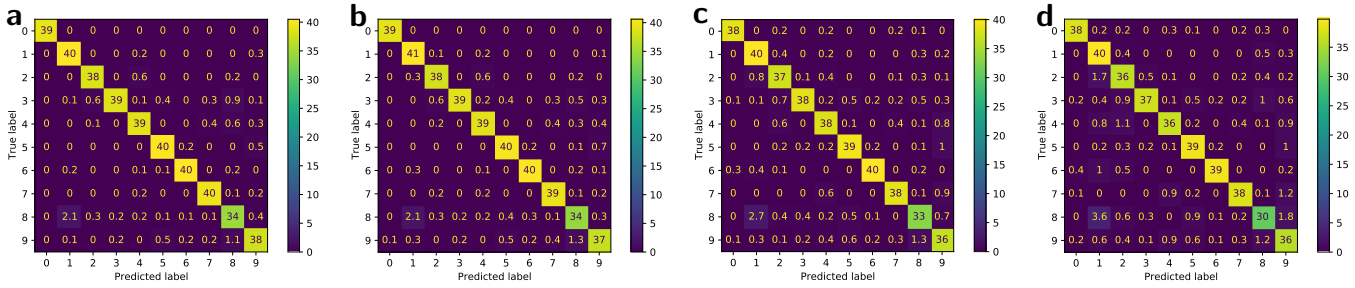

FIG. 9. Confusion matrix for the NPQC for **a)** radial basis function kernel **b)** exact quantum kernel **c)** mitigated IBM results **d)** unmitigated IBM results. We use 1300 training data and 200 test data, where we average the confusion matrix over 100 randomly sampled instances of the data.

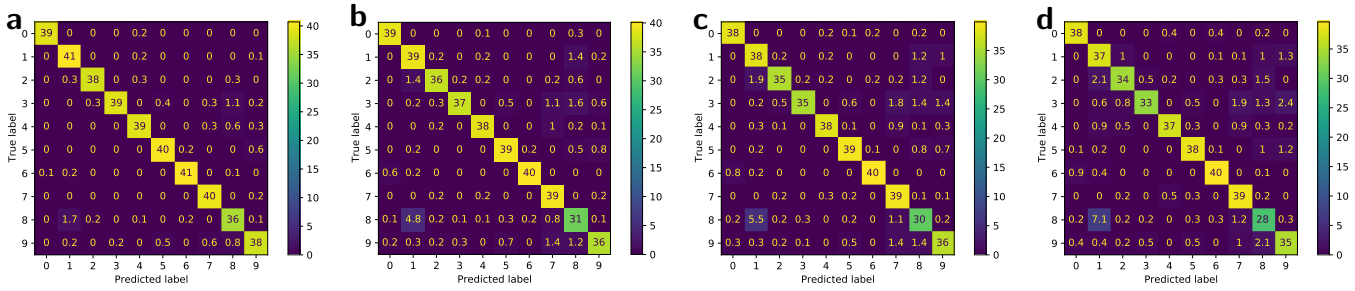

FIG. 10. Confusion matrix for the YZ-CX PQC for **a)** radial basis function kernel **b)** exact quantum kernel **c)** mitigated IBM results **d)** unmitigated IBM results. We use 1300 training data and 200 test data, where we average the confusion matrix over 100 randomly sampled instances of the data.