

Neural quantum kernels: Training quantum kernels with quantum neural networks

Pablo Rodriguez-Grasa^{1,2,3,*}, Yue Ban^{4,5,3} and Mikel Sanz^{1,2,6,7}

¹Department of Physical Chemistry, *University of the Basque Country UPV/EHU*, Apartado 644, 48080 Bilbao, Spain

²EHU Quantum Center, *University of the Basque Country UPV/EHU*, Apartado 644, 48080 Bilbao, Spain

³TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain

⁴Instituto de Ciencia de Materiales de Madrid (CSIC), Cantoblanco, E-28049 Madrid, Spain

⁵Departamento de Física, *Universidad Carlos III de Madrid*, Avda. de la Universidad 30, 28911 Leganés, Spain

⁶IKERBASQUE, Basque Foundation for Science, Plaza Euskadi 5, 48009 Bilbao, Spain

⁷Basque Center for Applied Mathematics (BCAM), Alameda de Mazarredo, 14, 48009 Bilbao, Spain



(Received 15 January 2024; accepted 20 May 2025; published 16 June 2025)

Quantum and classical machine learning have been naturally connected through kernel methods, which have also served as proof-of-concept for quantum advantage. Quantum embeddings encode classical data into quantum feature states, enabling the construction of embedding quantum kernels (EQKs) by measuring vector similarities and projected quantum kernels (PQKs) through projections of these states. However, in both approaches, the model is influenced by the choice of the embedding. In this work, we propose using the training of a quantum neural network (QNN) to construct neural quantum kernels, specifically neural EQKs and neural PQKs—problem-inspired kernel functions. Unlike previous approaches, our method requires the kernel matrix to be constructed only once, significantly reducing computational overhead. To achieve this, we introduce a scalable training method for an n -qubit data reuploading QNN. Furthermore, we demonstrate neural quantum kernels can alleviate exponential concentration and enhance generalization capabilities compared to problem-agnostic kernels, positioning them as a scalable and robust solution for quantum machine learning applications.

DOI: [10.1103/xphb-x2g4](https://doi.org/10.1103/xphb-x2g4)

I. INTRODUCTION

Quantum computing is a promising computational paradigm for tackling some complex computational problems which are classically intractable. In particular, its potential in enhancing machine learning tasks has garnered significant attention [1–4] with parametrized quantum circuits as the most common approach [5–8]. Although there are evidences of quantum advantage in some tailored problems [9–12], the advantage over classical counterparts for practical applications remains as an area of active research.

Amidst the exploration of quantum machine learning models, previous studies, particularly in Refs. [13,14], have delineated a categorization into explicit and implicit models. In explicit models, data undergoes encoding into a quantum state, and then a parametrized measurement is performed. In contrast, implicit or kernel models are based on a weighted summation of inner products between encoded data points. A specialized category within parametrized quantum circuits, which can be considered separately, consists of data reuploading models [15]. This architecture features an alternation between encoding and processing unitaries, yielding expressive models that have found extensive use [16–18].

Quantum kernel methods have garnered significant attention, both for evaluating their performance [19–25] and for theorizing their role in explaining quantum machine learning models [14,26,27]. This interest stems from several key factors. First, embedding data into quantum states provides access to the exponentially large Hilbert space, enabling efficient computation of inner products. Second, quantum feature maps facilitate the construction of quantum kernels, which may exhibit classical intractability and hold promise for quantum advantage [28]. Third, kernel methods, unlike neural networks, solve learning tasks through convex optimization in high-dimensional feature spaces, ensuring optimal solutions. This aligns with the representer theorem [29], which guarantees that kernel methods achieve a training loss lower than or equal to explicit models, given the same encoding and data set. However, this enhanced expressivity can sometimes compromise generalization, as discussed in Ref. [30].

However, quantum kernel methods present several challenges. Firstly, the computational complexity of constructing the kernel matrix scales quadratically with the number of training samples. Additionally, selecting the appropriate embedding that defines the kernel function is problem-dependent and requires careful consideration [31–33]. As highlighted in Ref. [34], building meaningful quantum machine learning models necessitates incorporating problem-specific knowledge. This can include, for instance, exploiting information about symmetries [35–39], among other data properties. This approach helps mitigate a key issue inherent in problem-agnostic methods: the exponential concentration of kernel values [40], which is particularly pronounced in embedding

*Contact author: pablojesus.rodriguez@ehu.eus

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International](https://creativecommons.org/licenses/by/4.0/) license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

quantum kernels (EQKs), where kernels are constructed by calculating inner products between large quantum states. In this context, a different class of quantum kernels known as projected quantum kernels (PQKs) [19] generates kernels through projections rather than full inner product calculations, thus alleviating the exponential concentration problem. However, an effective strategy is still required to design the appropriate embedding for these kernels.

When prior information about the problem is unavailable for designing a well-tuned embedding, problem-informed embeddings can be constructed through optimization. Two main approaches have been considered for this purpose: multiple kernel learning and kernel target alignment. In the former, a combination of kernel functions is optimized to minimize an empirical risk function [41,42]. In the latter, a kernel is defined using an embedding ansatz that is trained to align with an ideal kernel [43]. However, both methods necessitate recomputing the kernel matrix at each training iteration, leading to significant computational costs.

In this article, we propose the use of a quantum neural network (QNN) to construct neural quantum kernels. This approach allows for training the embedding that generates the kernel, requiring the kernel matrix to be constructed only once. Since training a QNN is not a straightforward task, we introduce a novel approach to scale the training of a data reuploading QNN to n qubits. This construction which has been already tested in a real-world classification problem [44], aims to circumvent trainability issues and clarifies the role of entanglement in accuracy, a factor that was previously unclear in most methods [45]. We demonstrate how our approach can be applied to both EQKs, identifying two specific cases: the 1-to- n approach, where a single-qubit neural network constructs an EQK for n qubits, and the n -to- n approach, where the neural network directly serves as the embedding to build the kernel, as well as to PQKs, thus providing broad applicability. Through numerical experiments, we demonstrate the effectiveness of our proposal by assessing not only model performance but also trainability and generalization ability, highlighting the importance of pretraining and the robustness of our method. Additionally, we evaluate its performance under noisy conditions.

II. QUANTUM KERNEL METHODS

Kernel methods are machine learning models defined by the linear combination

$$f_{\alpha,X}(\mathbf{x}) = \sum_{i=1}^M \alpha_i k(\mathbf{x}, \mathbf{x}_i), \quad (1)$$

where $\{\mathbf{x}_i\}_{i=1}^M$ represents training points from a set X . The kernel function $k(\mathbf{x}, \mathbf{x}_i)$ serves as a similarity measure, which must be symmetric and positive semidefinite. According to Mercer's Theorem, kernels can be estimated by evaluating the inner product of feature vectors in a Hilbert space $\mathcal{H}$, i.e., $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$. These feature vectors are obtained by applying a feature map ϕ to a data point $\mathbf{x}$. The parameters α are determined by solving a convex optimization problem. While there are proposals for solving this type of optimization problem on a quantum device [46,47], we

assume it is performed on a classical computer. This requires constructing the kernel matrix K , where each entry is given by $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Constructing K involves $O(M^2)$ inner product evaluations between the feature vectors.

In quantum kernel methods, feature vectors are quantum states, and the feature space is the Hilbert space where these states reside. While other constructions, such as projected quantum kernels [19], are possible, most quantum kernel constructions focus on EQKs, which are proven to be universal [48]. EQKs are defined by using a quantum embedding that encodes data into quantum states $\rho(\mathbf{x})$, implicitly defining the EQK

$$k_{ij} = \text{tr}(\rho(\mathbf{x}_i) \rho(\mathbf{x}_j)). \quad (2)$$

For pure states, the quantum embedding $S(\cdot)$ defines the quantum feature states on n qubits as $\rho(\mathbf{x}) = S(\mathbf{x})|\mathbf{0}\rangle\langle\mathbf{0}|S(\mathbf{x})^\dagger$, where $|\mathbf{0}\rangle \equiv |0\rangle^{\otimes n}$. Thus, the kernel simplifies to

$$k_{ij} = |\langle\mathbf{0}|S(\mathbf{x}_i)^\dagger S(\mathbf{x}_j)|\mathbf{0}\rangle|^2. \quad (3)$$

This quantity can be estimated on a quantum computer through various methods [49–51]. One approach involves preparing the state $S^\dagger(\mathbf{x}_i)S(\mathbf{x}_j)|\mathbf{0}\rangle$ and measuring the probability that all qubits are in the $|\mathbf{0}\rangle$ state.

An important insight from classical machine learning theory is captured by the representer theorem. The quantum version of this theorem asserts that, given a quantum embedding and a training set, models in the form of Eq. (1) consistently achieve training losses that are equal to or lower than those of explicit models, which take the form

$$f_{\theta}(\mathbf{x}) = \text{tr}(\rho(\mathbf{x}) O_{\theta}), \quad (4)$$

where O_{θ} is a parametrized observable that can be tuned. Given the representer theorem, the proof of concept that quantum embeddings can lead to quantum advantages [28], and the inherent capability of quantum devices to access exponentially large feature spaces, quantum kernel methods emerge as promising candidates for quantum machine learning models [26]. They also offer provable guarantees and flexibility, similar to training neural networks with large hidden layers, which is equivalent to using neural tangent kernels [33]. However, accessing such large spaces is associated with the issue of exponentially vanishing kernel values [34,40,52–54]. As studied in Ref. [40], problem-inspired embeddings can mitigate this issue compared to problem-agnostic embeddings. Therefore, designing suitable quantum embeddings tailored to the problem is crucial. When information about the data, such as symmetries [37–39,55], is available, it can be leveraged to construct an appropriate quantum embedding. For example, Ref. [28] demonstrates a classification task based on the discrete logarithm problem, using a quantum embedding inspired by Shor's factorization algorithm. However, in the absence of such information, quantum kernel training becomes particularly useful. The common strategy involves employing a parameterized quantum embedding $S_{\boldsymbol{\gamma}}(\cdot)$ [56], where $\boldsymbol{\gamma}$ represents the trainable parameters. This embedding defines a trainable quantum kernel $k_{\boldsymbol{\gamma}}(\mathbf{x}_i, \mathbf{x}_j)$, which is then optimized based on a specific figure of merit. For EQKs, these kernel training methods are based on multiple kernel learning, aiming to determine the optimal combination of different kernels [41,42], and kernel target alignment [43], where

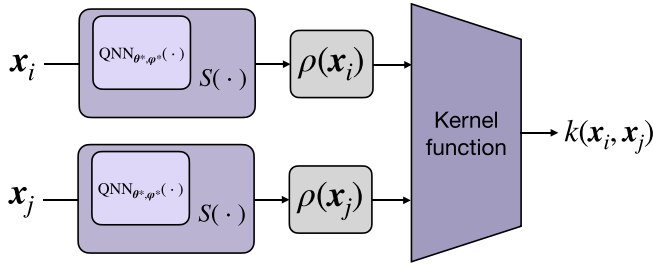


FIG. 1. Illustration of neural quantum kernels. Using a trained quantum neural network $\text{QNN}_{\theta^*, \varphi^*}(\cdot)$ we construct a quantum embedding $S(\cdot)$ which generates quantum feature states $\rho(\cdot)$. By applying a kernel function to pairs of these quantum feature states, we obtain the quantum kernel $k(x_i, x_j)$. Depending on the nature of this kernel function, we can construct either neural EQKs or neural PQKs.

the embedding is trained to resemble an ideal kernel matrix with maximum overlap between quantum states representing the same class and minimum overlap for states of different classes. However, these approaches require constructing the kernel matrix at each training step, which implies a high computational cost. Additionally, the kernel target alignment strategy suffers from barren plateaus in the optimization [40].

In this work, we introduce a novel method for training quantum kernels that constructs the kernel matrix only once and operates outside the regime where barren plateaus are theoretically expected [57–65]. The method first trains a QNN, for which we propose a data reuploading architecture to enable efficient training and scalability. Once trained, we leverage the learned parameters to construct different types of quantum kernel (see Fig. 1 for an illustration of this process).

III. SCALING DATA REUPLOADING FOR n -QUBIT QNN

We propose constructing neural quantum kernels by training data reuploading QNNs. A scalable training approach, starting with a smaller QNN and progressively extending it, reduces computational cost, improves convergence, and enables practical large-scale implementations, ensuring feasibility for quantum kernels as system sizes grow.

As reported by Pérez-Salinas *et al.* in Ref. [15], the data reuploading model incorporates layers composed of data-encoding and training unitaries. This approach effectively introduces nonlinearities to the model allowing to capture complex patterns on data [16,17,66,67]. In fact, it has been demonstrated that a single qubit quantum classifier possesses universal capabilities [68].

While various encoding strategies could be considered for this architecture, we specifically adopt the easiest one defining

$$\begin{aligned} \text{QNN}_{\theta}(\mathbf{x}) &\equiv \prod_{l=1}^L U(\theta_l) U(\mathbf{x}) \\ &= U(\theta_L) U(\mathbf{x}) \dots U(\theta_1) U(\mathbf{x}). \end{aligned} \quad (5)$$

Here, L denotes the number of layers, U represents a generic $\text{SU}(2)$ unitary, and the vector $\theta = \{\theta_1, \dots, \theta_L\}$ encompasses the trainable parameters. To leverage this model to construct a binary classifier, one must select two label states that are

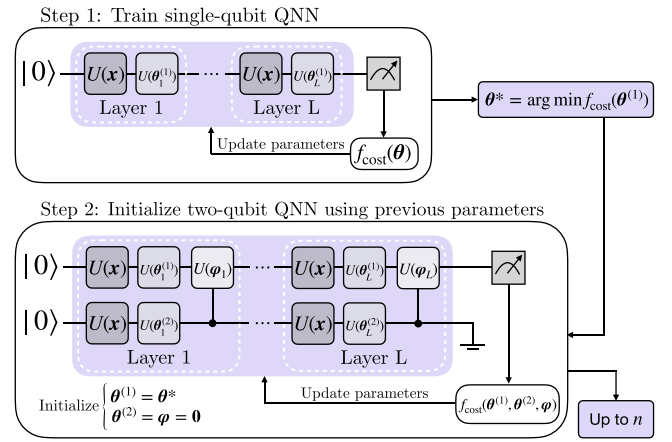


FIG. 2. Iterative training of a two-qubit data reuploading QNN. In Step 1, a single-qubit QNN is trained to obtain optimal model parameters denoted as θ^* . Moving to Step 2, training for the two-qubit QNN is initiated, initializing new extra parameters to 0, while the parameters of the first qubit are set to θ^* . This iterative approach can scale the QNN up to n qubits, ensuring that the n -qubit QNN performs at least as effectively as the $n - 1$ version.

maximally separated in Hilbert space. The training objective involves instructing the model to collectively rotate points belonging to the same class, bringing them closer to their corresponding label state.

Starting with the data reuploading single-qubit QNN architecture, we can naturally extend it to create multiqubit QNNs. The introduction of more qubits enhances the model's expressivity by increasing the number of trainable parameters per layer and offering the potential for entanglement [69].

In this work, we propose an iterative training approach for multiqubit QNNs. In our construction, the n -qubit QNN is defined as

$$\text{QNN}_{\theta, \varphi}(\mathbf{x}) = \prod_{l=1}^L \left(\prod_{s=1}^{n-1} \text{CU}_{s+1}^s(\varphi_l^{(s)}) \left(\bigotimes_{r=1}^n U(\theta_l^{(r)}) \right) U(\mathbf{x})^{\otimes n} \right), \quad (6)$$

where $\prod_{i=1}^n A_i = A_n \dots A_2 A_1$ and CU_{s+1}^s denotes the controlled version of the general $\text{SU}(2)$ unitary with control in the $(s + 1)$ -th qubit and target in the s -th qubit, and θ and φ refer to the trainable parameters of single-qubit and two-qubit gates, respectively. The total number of trainable parameters in this architecture is $3(2n - 1)L$.

To train the n -qubit QNN, we propose an iterative construction starting from a single-qubit QNN, as shown in Fig. 2. Initially, we train a single-qubit QNN and use its parameters to initialize the two-qubit QNN. For the two-qubit QNN initialization, we set $\varphi_l^{(1)} = \mathbf{0}$ for all $l \in [1, L]$, keeping the parameters of the first qubit from the single-qubit training step. As a result, the entangling layers have no effect, so the output state of the first qubit at the beginning of the training matches that of the single-qubit QNN. This method allows us to scale the architecture, adding qubits incrementally. When adding a new qubit, we initialize the entangling gates as identities, and training starts with the optimal parameters from the previous step. This structured approach enables the systematic

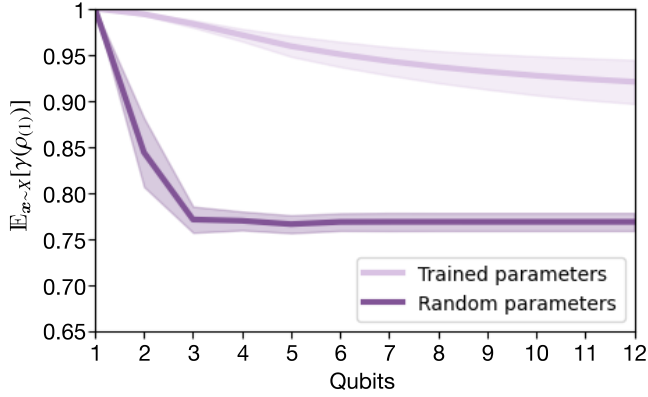


FIG. 3. Mean and standard deviation of the mean purity of the reduced density matrix for the first qubit, calculated over five distinct random DATA SETS, each containing 50 data points from the Fashion MNIST data set. The mean purity is plotted for both the proposed architecture trained iteratively and the same architecture with randomly initialized parameters.

and scalable enhancement of the QNN's performance as more qubits are added.

Our strategy addresses key causes of barren plateaus. One major factor is the use of global cost functions; we mitigate this by using a local cost function. Another factor is entanglement: even considering a local measurement, highly entangled states can lead to barren plateaus. In our method, entanglement is introduced gradually. As shown in Fig. 3, the resulting state remains only moderately entangled as additional qubits are added. In this figure, we plot the mean purity of the first qubit $\gamma(\rho_{(1)})$, defined as

$$\mathbb{E}_{x \sim X}[\gamma(\rho_{(1)})] = \mathbb{E}_{x \sim X}[\text{tr}(\rho_{(1)}^2(x))] = \frac{1}{M} \sum_{i=1}^M \text{tr}(\rho_{(1)}^2(x_i)). \quad (7)$$

If the state were highly entangled, the single-qubit purity would approach 1/2, the value corresponding to a maximally mixed state. However, as shown in Fig. 3, the purity decreases only slightly with increasing system size and remains well above this value, indicating that entanglement is introduced

in a structured and controlled manner. To further support this point, we compare the trained QNN to a scenario where the parameters are randomly initialized rather than optimized iteratively. In this case, the reduced state becomes more mixed, suggesting that the randomization leads to the entanglement growth and a more uniform Hilbert space exploration. The structured entanglement observed in the trained QNN helps preserve gradient magnitudes, mitigating the barren plateau problem. To generate the figure, we considered five randomly sampled data sets X , each containing 50 samples from the Fashion MNIST data set.

IV. NEURAL EMBEDDING QUANTUM KERNELS

Considering a parameterized quantum embedding $S_{\gamma}(\cdot)$, in neural EQKs γ are the parameters obtained from training a QNN. In the following subsections, we present two specific cases of neural EQKs: the n -to- n and the 1-to- n configurations.

A. n -to- n approach

In this construction, an n -qubit QNN is trained using the iterative method proposed in Sec. III to directly construct the corresponding EQK of n qubits. As depicted in Fig. 4, a multiqubit QNN of the form given by Eq. (6) is trained while fixing the trainable parameters of the embedding θ^* and φ^* . These parameters are then used to construct a quantum embedding

$$S(\cdot) = \text{QNN}_{\theta^*, \varphi^*}(\cdot), \quad (8)$$

which defines the corresponding EQK as given by Eq. (3).

This method allows us to scale the QNN as much as possible during training. Once it reaches a performance plateau, we can utilize the trained feature map to construct an EQK.

Using the representer theorem it can be shown that the kernels derived from the QNN will be at least as effective as the corresponding QNN. In our construction, the n -qubit QNN is defined in Eq. (6). There, we can extract the last layer to separate it into a variational part, which will be absorbed into the measurement when we define the corresponding quantum model, and the encoding part which will define the quantum feature map for the construction of the kernel. This corresponds to

$$\text{QNN}_{\theta, \varphi}(x) = \underbrace{\prod_{s=1}^{n-1} \text{CU}_{s+1}^s(\lambda^{(s)}) \left(\bigotimes_{r=1}^n U(\omega^{(r)}) \right)}_{=V(\lambda, \omega)} \times \underbrace{\prod_{l=1}^{L-1} \left(\prod_{s=1}^{n-1} \text{CU}_{s+1}^s(\varphi_l^{(s)}) \left(\bigotimes_{r=1}^n U(\theta_l^{(r)}) \right) U(x)^{\otimes n} \right)}_{=S_{\theta, \varphi}(x)}. \quad (9)$$

where we defined $\lambda \equiv \varphi_L$ and $\omega \equiv \theta_L$, which are the parameters of the last layer. This formulation aligns with the definition of a quantum model from Ref. [26],

$$f(x) = \text{tr}(\rho(x)\mathcal{M}). \quad (10)$$

In our case,

$$\rho(x) := \rho_{\theta, \varphi}(x) = S_{\theta, \varphi}(x)|0\rangle\langle 0|^{\otimes n} S_{\theta, \varphi}(x)^\dagger, \quad (11)$$

and the variational measurement

$$\mathcal{M} := \mathcal{M}_{\lambda, \omega} = V(\lambda, \omega) (\hat{\sigma}_z \otimes \mathbb{1}^{(n-1)}) V(\lambda, \omega)^\dagger. \quad (12)$$

Once this model is trained over a data set, we determine the parameters that minimize the f_{cost} defined in the previous section, fixing β^* , θ^* , λ^* , and ω^* . If we now use the corresponding feature map S_{θ^*, φ^*} to construct an EQK, we are effectively replacing the measurement from the optimization

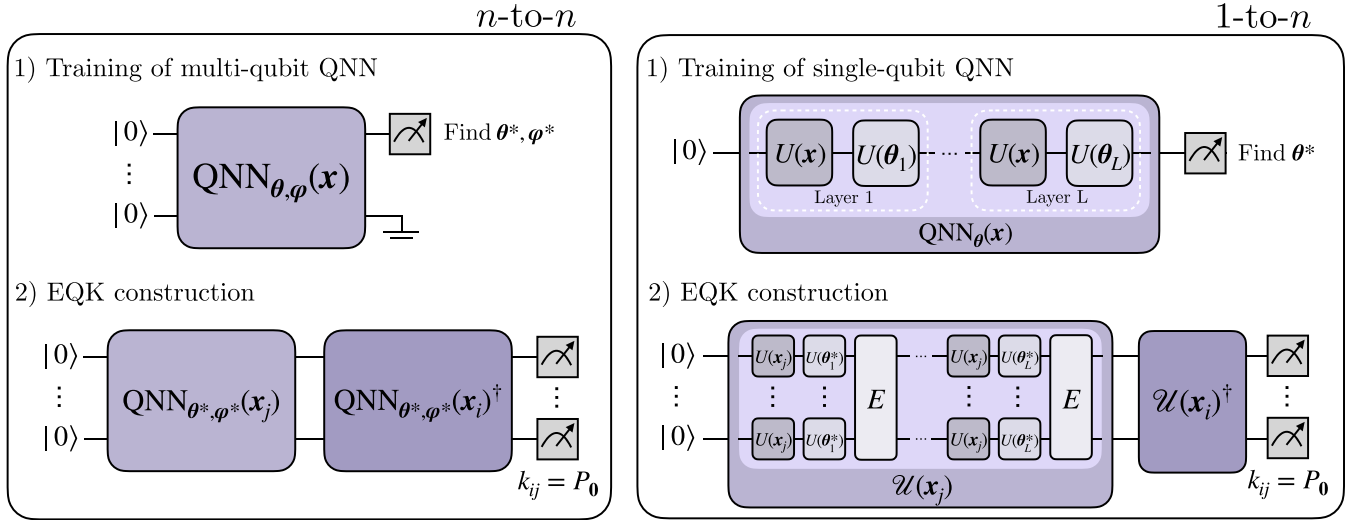


FIG. 4. Neural embedding quantum kernels. On the left, we have the n -to- n proposal, constructed by directly utilizing the trained data reuploading n -qubit QNN as the quantum embedding. On the right, we show the construction of an embedding quantum kernel from the training of a single-qubit QNN, named as 1-to- n . The kernel matrix element k_{ij} is defined as the probability of measuring all qubits in the state $|0\rangle$, denoted as P_0 .

$\mathcal{M}_{\lambda^*, \omega^*}$ by the optimal measurement

$$\mathcal{M}_{\text{opt}} = \sum_{m=1}^M \alpha_m \rho_{\theta^*, \varphi^*}(\mathbf{x}_m), \quad (13)$$

which, by the representer theorem, defines the optimal quantum model

$$f_{\text{opt}}(\mathbf{x}) = \sum_{m=1}^M \alpha_m \text{tr}(\rho_{\theta^*, \varphi^*}(\mathbf{x}) \rho_{\theta^*, \varphi^*}(\mathbf{x}_m)) \quad (14)$$

that minimizes the regularized empirical risk function. Thus, we proved that constructing the EQK from QNN training will perform equally or better in terms of training loss than the QNN alone.

This formal relationship with the representer theorem can be intuitively seen in Fig. 5, specifically for a single-qubit scenario. Initially, the QNN rotates data points of the same class near their label state while maintaining the decision hyperplane fixed. This hyperplane is taken as the equator of the Bloch sphere, i.e., $\langle \hat{\sigma}_z \rangle = 0$. After training the QNN, the resulting feature map is obtained by preserving the parameters acquired during training. This feature map is then utilized to construct an EQK. The subsequent application of the support vector machine (SVM) algorithm, using this kernel, aims to identify the optimal separation hyperplane in the feature space. In this context, the optimization is equivalent to adjusting the optimal measurement $\mathcal{M}(\gamma)$ while keeping the data points fixed.

Therefore, the QNN does not need to be perfectly trained; even if the points of the same class do not align exactly with their corresponding label states, the optimal measurement derived from the kernel construction can effectively separate the two classes.

B. 1-to- n approach

A more nontrivial approach is the 1-to- n construction, where we train a single-qubit QNN $_{\theta}$ from Eq. (5), fixing the θ^* and we leverage this training to construct an EQK given by

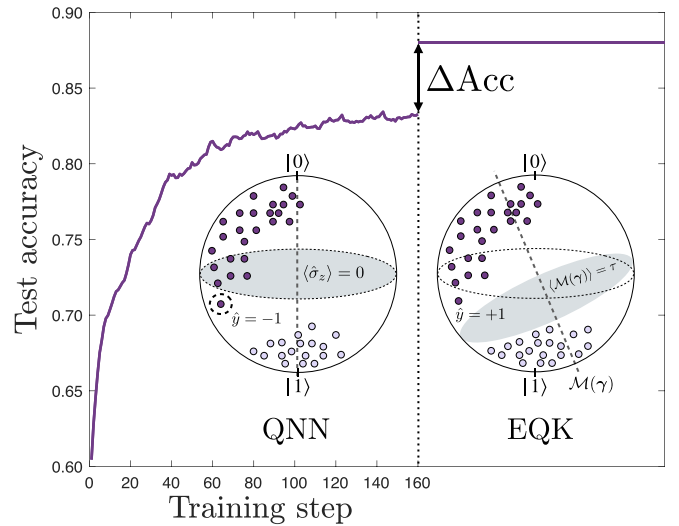


FIG. 5. An schematic illustration of how a single-qubit EQK, constructed from a single-qubit QNN, can enhance classification outcomes. The QNN aims to group points of the same class while fixing the decision plane (left Bloch sphere). The SVM, employing the single-qubit EQK, fine tunes the decision boundary parameters to find the optimal hyperplane (right Bloch sphere). As a result, data points that were previously misclassified can now be correctly assigned to their respective labels. We also provide an example demonstrating a scenario in which the QNN accuracy plateaus out for a specific data set. By using the corresponding EQK, we obtain however an increase in accuracy, denoted as ΔAcc .

the embedding

$$S(\cdot) = \prod_{l=1}^L E U(\theta_l^*)^{\otimes n} U(\cdot)^{\otimes n}, \quad (15)$$

where E denotes an entangling operation, such as a cascade of CNOT or CZ gates, among other possibilities. It is worth noting that the training is conducted for a single-qubit QNN and does not explicitly consider entanglement. Nevertheless, we will present numerical results demonstrating that this training alone is sufficient to select parameters for constructing a customized multiqubit EQK tailored to a specific task. Remarkably, the training of the single-qubit QNN could be executed on a classical computer, and subsequently, the derived parameters could be utilized to construct a potent kernel on a quantum computer.

Certainly, one can combine both the n -to- n and the 1-to- n architectures and generalize it to n -to- $m \cdot n$, where m represents some integer. In this scenario, an n -qubit QNN is trained and utilized to implement the same design as in the 1-to- n construction. However, in this case, each qubit of the QNN is embedded into m qubits, introducing entanglement between layers.

V. NEURAL PROJECTED QUANTUM KERNELS

Given a quantum embedding, instead of constructing a quantum kernel directly from the inner product of the full quantum feature states, Ref. [19] introduced PQKs. This approach builds the kernel using projections of these quantum feature states.

An example of such kernels is

$$k_{ij} = \exp \left(-\gamma \sum_k \|\rho^{(k)}(\mathbf{x}_i) - \rho^{(k)}(\mathbf{x}_j)\|_F^2 \right), \quad (16)$$

where $\rho^{(k)}(\mathbf{x}_i) = \text{tr}_{i \neq k} \rho(\mathbf{x})$ represents the one-particle reduced density matrix on qubit k , γ is a tunable hyperparameter, and $\|\cdot\|_F$ denotes the Frobenius norm. However, these types of kernels still face the challenge of their performance being dependent on the choice of the quantum embedding that defines the quantum feature states $\rho(\mathbf{x})$.

We can construct problem-inspired PQKs using our method by pretraining the quantum embedding using a QNN. Specifically, we consider a simpler PQK construction given by

$$k_{ij}^{PQ} = \text{tr}(\rho_{(1)}(\mathbf{x}_i) \rho_{(1)}(\mathbf{x}_j)), \quad (17)$$

where the quantum embedding is defined by a trained n -qubit QNN. This approach allows us to define a neural PQK by setting

$$\rho_{(1)}(\mathbf{x}) = \text{tr}_{j \neq 1} (\text{QNN}_{\theta^*, \varphi^*}(\mathbf{x}) |0\rangle \langle 0| \text{QNN}_{\theta^*, \varphi^*}(\mathbf{x})^\dagger), \quad (18)$$

which is depicted in Fig. 6. In Sec. VI, we demonstrate the importance of pretraining the parameters θ and φ . We compare the performance of the neural PQK with that of a PQK constructed from a randomly fixed quantum embedding, highlighting the benefits of our proposed approach.

As discussed in Ref. [40], PQKs benefit from reduced exponential concentration behaviors. Additionally, incorporating pretraining into the quantum embedding introduces an

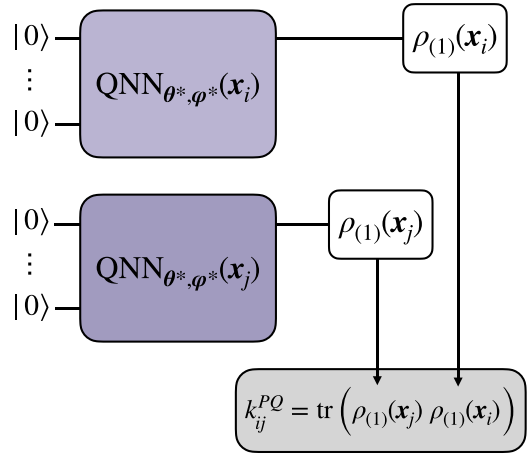


FIG. 6. Neural projected quantum kernels. Utilizing the trained quantum neural network $\text{QNN}_{\theta^*, \varphi^*}(\cdot)$, we construct a projected quantum kernel by computing the trace between reduced quantum feature states, specifically using the reduced density matrix on the first qubit $\rho_{(1)}(\cdot)$.

inductive bias towards the considered data set. This bias can reduce the need for high expressivity, potentially requiring fewer layers to achieve the same performance.

VI. NUMERICAL RESULTS

In this section, we present numerical results to validate the proposed neural quantum kernels. Specifically, we tackle a binary classification problem on the Fashion MNIST data set [70], with the objective of distinguishing between dresses (class 3) and shirts (class 6). To reduce the dimensionality, we applied principal component analysis, compressing the feature space to three components—corresponding to the number of features that can be encoded using a single encoding unitary. The evaluation includes both ideal and noisy simulations, utilizing 500 training samples and 300 test samples.

The QNN architecture consists of $L = 7$ layers. For training, we use the Adam optimizer with a batch size of 24. During the first training step $n = 1$, the learning rate is set to 0.05, and the model is trained for 30 epochs. For subsequent steps $n > 1$, the learning rate is reduced to 0.005, and training is conducted for 10 epochs.

For the noisy simulations, we applied two single-qubit quantum channels: amplitude damping and phase damping, each following the application of every quantum gate. Using the Kraus decomposition to represent the action of the noise channels

$$\rho \rightarrow \sum_i K_i \rho K_i^\dagger, \quad (19)$$

the amplitude damping channel is described by the Kraus operators

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix}, \quad (20)$$

$$K_1 = \begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix}, \quad (21)$$

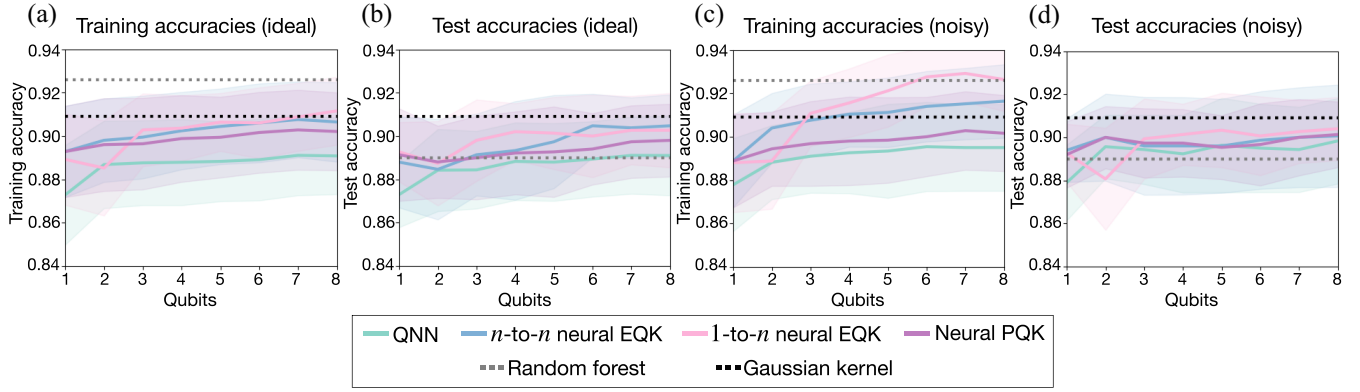


FIG. 7. Numerical results for the binary classification problem on the Fashion MNIST data set. Training and test accuracies, averaged over five independent experiments with different random seeds, are reported for both ideal [(a) and (b)] and noisy [(c) and (d)] simulations. The results show the mean and standard deviation of the QNN performance using our scaling construction, as well as the three proposed neural quantum kernels.

where $\gamma \in [0, 1]$ represents the amplitude damping probability, which can be defined as $\gamma = 1 - e^{-\Delta t/T_1}$, with T_1 being the thermal relaxation time and Δt the duration of the quantum gate application. For the phase damping channel, the Kraus operators are

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\lambda} \end{pmatrix}, \quad (22)$$

$$K_1 = \begin{pmatrix} 0 & 0 \\ 0 & \sqrt{\lambda} \end{pmatrix}, \quad (23)$$

where $\lambda \in [0, 1]$ is the phase damping probability, which is given by $\lambda = 1 - e^{-\Delta t/T_2}$, and T_2 denotes the dephasing time.

Considering the experimental values for the noisy parameters of a superconducting quantum processor, which are $50 \mu\text{s} \leq T_1 \leq 150 \mu\text{s}$, $25 \mu\text{s} \leq T_2 \leq 75 \mu\text{s}$, and $10 \text{ ns} \leq \Delta t \leq 50 \text{ ns}$, we note that the noise level increases when T_1 and T_2 decrease and Δt increases. For our numerical experiments, we considered the worst-case scenario for these parameters: $T_1 = 50 \mu\text{s}$, $T_2 = 25 \mu\text{s}$, and $\Delta t = 90 \text{ ns}$.

Figure 7 illustrates the training and test accuracies for the different neural quantum kernel models proposed under both ideal and noisy simulation scenarios. The results represent the mean accuracy across five data set samples. For the 1-to-n approach, the entangling operation is implemented using a cascade of CNOT gates. Taking Eq. (15), this means considering

$$E = \prod_{s=1}^{n-1} \text{CNOT}_s^{s+1}. \quad (24)$$

Although kernel models are prone to overfitting, we observe that in this case, the training accuracies [Figs. 7(a) and 7(c)] are only slightly higher than the test accuracies [Fig. 7(b) and 7(d)], highlighting the notable generalization ability of the models. This disparity is more pronounced in the noisy scenario [Fig. 7(c) and 7(d)], as expected, since noise reduces the model's generalization capability. In all cases, accuracies improve as the number of qubits increases, driven by the iterative construction of the QNN. However, this improvement is less pronounced for the test accuracies in the noisy scenario.

The QNN's performance is generally lower than that of kernel-based models. While this aligns with the representer theorem, which applies to training accuracies, we also observe that neural quantum kernels outperform the QNN in test accuracies as well. Although the QNN achieves competitive performance on its own, our proposal implies using it as a foundation for constructing quantum kernels, enabling more powerful models. However, as the kernel construction process requires additional quantum computational steps, its advantage diminishes in the presence of noise due to error propagation. Nonetheless, neural quantum kernels consistently outperform the standalone QNN, even in noisy conditions.

In the ideal scenario, especially for test accuracies, the neural EQK models tend to outperform neural PQQ models. However, this performance gap narrows in the noisy scenario, as neural PQQ models use circuits with half the depth of neural EQK models, making them more robust to noise.

Furthermore, Fig. 8 highlights the critical role of pre-training in constructing neural PQQs. It compares the test accuracies of our proposed neural PQQ approach with those obtained without pretraining, where random parameters are used to create a problem-agnostic quantum embedding, referred to as random PQQ. Specifically, the parameters θ and ϕ are initialized randomly, following a uniform distribution in the range $[0, 1]$. The substantial difference in performance underscores the effectiveness of our neural quantum kernel strategy.

To compare the quantum models with classical methods, we employed two classical machine learning approaches as benchmarks: a SVM with a Gaussian kernel and a random forest. For both methods, a grid search was conducted over the hyperparameters, and the optimal combination was selected using crossvalidation.

For the SVM model, the hyperparameters considered were the regularization parameter

$$C \in \{0.1, 1, 10, 100\}, \quad (25)$$

and the kernel hyperparameter

$$\gamma \in \{0.01, 0.1, 1, 10\}. \quad (26)$$

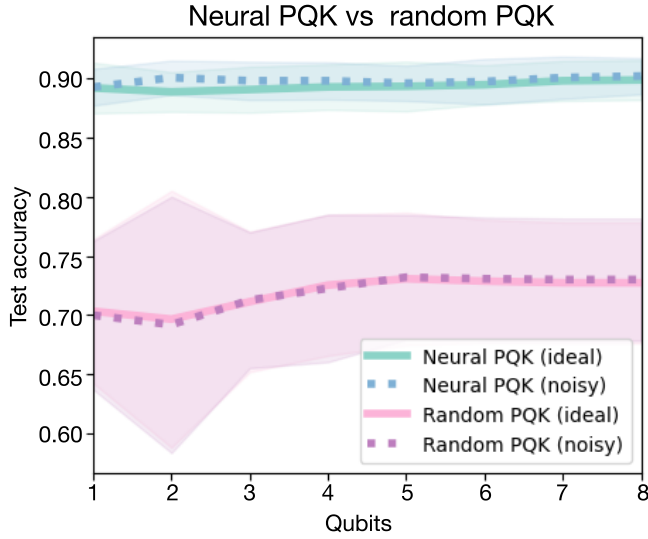


FIG. 8. Test accuracies for the proposed neural PQQ compared to a PQQ with random parameters, which is equivalent to using a problem-agnostic quantum embedding. The results are presented for both the ideal and noisy scenarios.

For the random forest model, the hyperparameters included the maximum depth of individual trees

$$\text{max\_depth} \in \{2, 3, 4, 5\}, \quad (27)$$

and the number of trees

$$\text{n\_estimators} \in \{25, 50, 100, 200, 500\}. \quad (28)$$

While the quantum models underperform these classical methods with just 1-2 qubits, increasing the number of qubits allows the quantum models to achieve competitive test accuracies. Additional numerical results are provided in Appendix E 3.

These findings demonstrate two key aspects: first, the scalability of a data reuploading QNN and the improved performance obtained by constructing the corresponding neural quantum kernel, and second the effectiveness of the 1-to- n approach in creating complex kernels by training simple QNN architectures. Although the small QNNs can be trained on classical computers, the resulting parameters can be used to construct an EQK inspired by classically hard problems, showcasing the potential for enhanced performance.

VII. TRAINABILITY AND GENERALIZATION CAPACITY

To build an effective quantum machine learning model, one that can make accurate predictions on unseen data, it must be both trainable and exhibit low generalization error.

Regarding trainability, it is important to distinguish between two types: the trainability of the embedding and the trainability that follows after the embedding is selected. In our case, the first refers to the training of the QNN, which involves optimizing a nonconvex cost function. The second type occurs after the embedding is chosen, where we use the kernel matrix as input and optimize the kernel model parameters by solving a convex optimization problem. In this scenario, with a fixed

embedding, the optimization method is guaranteed to find the optimal solution.

However, constructing the kernel matrix K involves sampling the matrix elements $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ on a quantum computer. This process relies on measurements (shots), and if the differences between kernel values are very small, a large number of shots will be required to accurately estimate them. Specifically, if the kernel matrix elements exhibit exponential concentration and we only have a polynomial number of shots, the resulting optimized model may become independent of the input data, undermining its ability to generalize effectively. Using Chebyshev's inequality, the concentration of the kernel values is expressed as

$$\Pr_{\mathbf{x}_i, \mathbf{x}_j} [|k_{ij} - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j} [k_{ij}]| \geq \delta] \leq \frac{\text{Var}_{\mathbf{x}_i, \mathbf{x}_j} [k_{ij}]}{\delta^2} \quad (29)$$

for any $\delta > 0$. Here, the probability and variance are evaluated over all possible pairs of input data. This differs from the exponential concentration in QNNs, where the variance is studied in relation to the trainable parameters. Specifically, if the variance decreases exponentially with the number of qubits,

$$\text{Var}_{\mathbf{x}_i, \mathbf{x}_j} [k_{ij}] \in O(c^{-n}), \quad (30)$$

for $c > 1$, the kernel values will concentrate exponentially around the mean. In this case, an exponential number of measurements will be required to accurately approximate the kernel matrix.

Just as highly expressive quantum ansätze can lead to barren plateaus in QNNs [59], highly expressive quantum embeddings can result in exponential concentration in quantum kernel methods. Specifically, both embedding and projected quantum kernels exhibit this exponential decay when the ensemble of unitaries, generated by applying the embedding to the training set, is exponentially close to a 2-design (see Theorem 1 in Ref. [40]). This underscores the importance of designing problem-inspired embeddings.

In Fig. 9, we present the scaling of the variance as a function of the number of qubits for different numbers of layers, comparing EQKs (k_{ij}^{EQ}) and PQQs (k_{ij}^{PQ}) in the neural construction versus problem-agnostic embeddings, where the embedding parameters are selected randomly. The variance is computed from the kernel matrix, excluding the diagonal terms for EQKs as these are equal to one. For EQKs, we observe that the variances for both the neural approach and random parameter selection are quite similar and do not exhibit exponential decay. In contrast, for PQQs with random parameters, the variances show an exponential decrease, particularly for $L = 100$ layers. Notably, while the behavior of PQQs with random parameters aligns with previous observations in Ref. [40], the neural construction yields a strikingly different trend: the variances remain stable as the number of qubits increases. This highlights the importance of pretraining the quantum embedding to mitigate the exponential concentration of kernel values.

On the other hand, the fact that a model is trainable does not guarantee its ability to make accurate predictions on new data, as quantum models can memorize random patterns [71]. Therefore, it is crucial to ensure that the model has a low generalization error, ϵ_{gen} , which is defined as the difference

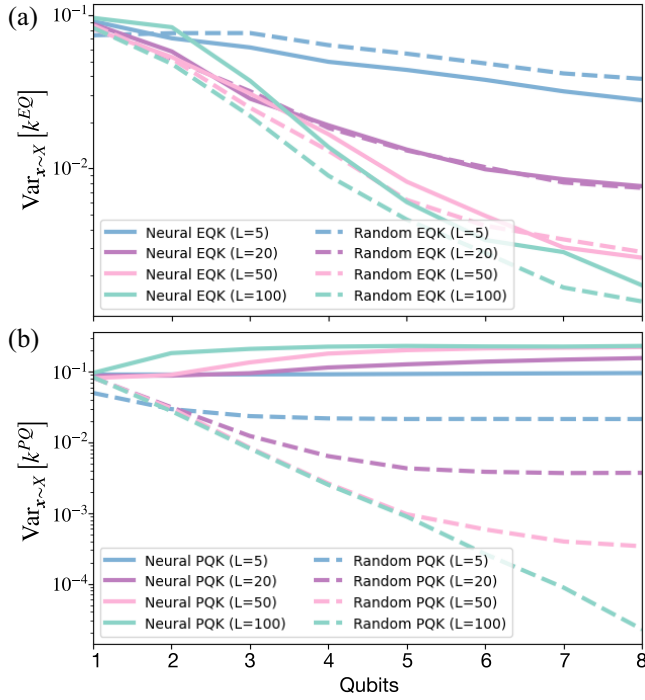


FIG. 9. Variances of the (a) embedding and (b) projected quantum kernels for both the neural construction and random parameters, plotted as a function of the number of qubits n and the number of layers L . The data used consists of 50 samples from the Fashion MNIST data set.

between the true error and the training error. For a training set of M data points $(x_i, y_i)_{i=1}^M$, this error can be upper-bounded with probability at least $1 - \delta$ as follows:

$$\epsilon_{\text{gen}} \leq O\left(\sqrt{\frac{\sum_{i=1}^M \sum_{j=1}^M A_{i,j} y_i y_j}{M}} + \sqrt{\frac{\log(1/\delta)}{M}}\right), \quad (31)$$

as demonstrated in Ref. [19]. Here, the matrix A is defined as

$$A = (K + \lambda \mathbb{1})^{-1} K (K + \lambda \mathbb{1})^{-1}, \quad (32)$$

where K represents the kernel matrix and λ is the regularization parameter. In Fig. 10, we plot the first term of ϵ_{gen} , referred to as the generalization error bound, as a function of the hyperparameter λ for $n = 8$ qubits. When $\lambda = 0$, the training error is zero, which corresponds to the maximum generalization error.

As shown in Fig. 10, employing the neural approach results in a lower generalization error bound compared to a problem-agnostic embedding with randomly chosen parameters. Furthermore, we observe that PQKs exhibit a lower generalization error bound overall. The difference between random and neural approaches is more pronounced in PQKs, whereas in EQKs, the advantage of the neural approach over the random one is less significant. These results represent the mean of five independent experiments conducted on the Fashion MNIST data set with $M = 50$ training samples.

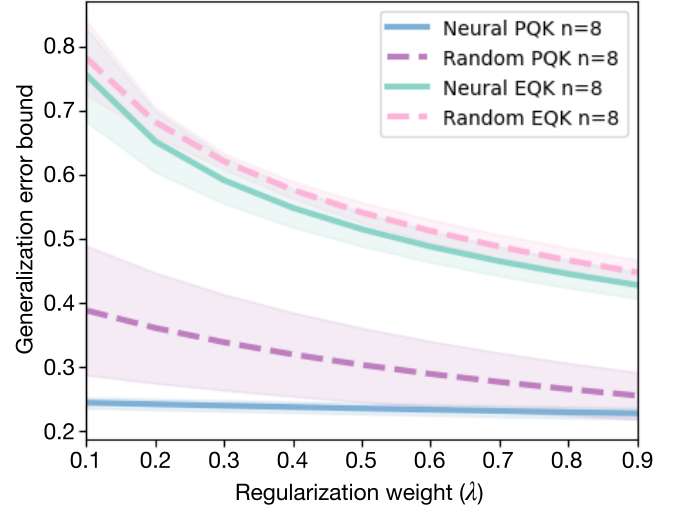


FIG. 10. Generalization error bound as a function of the regularization weight for both neural EQK and PQK, compared to random parameter selection for the embedding. All models use $n = 8$ qubits, with the neural EQK corresponding to the n -to- n approach. The data consists of 50 samples from the Fashion MNIST data set.

VIII. CONCLUSIONS

Building meaningful quantum machine learning models requires designing problem-inspired quantum embeddings. Neural quantum kernels enable the encoding of problem-specific information into quantum kernel methods by training a neural network, offering a more efficient alternative to previous approaches that required constructing the kernel matrix at every training step. Through our proposed method for scaling QNNs, we demonstrate how they can effectively generate powerful neural quantum kernels capable of competing with well-optimized classical machine learning algorithms. Specifically, we introduce neural EQKs with two distinct constructions and a structured approach for designing neural PQKs. Our results highlight the advantages of neural quantum kernels over problem-agnostic embeddings in terms of performance, trainability, and generalization error. While neural EQKs achieve higher accuracy, neural PQKs exhibit greater robustness in noisy scenarios, as well as improved trainability and generalization. Although there are no formal guarantees that trainable QNNs yield classically hard or non-trivial embeddings, our focus lies in the empirical scalability of quantum models. In practice, achieving strong performance often requires scaling to regimes where classical methods may become inefficient, thus offering a practical pathway toward classical intractability. These findings pave the way for developing scalable and resilient quantum kernel models, with potential extensions to quantum-inspired methods.

ACKNOWLEDGMENTS

The authors thank M. Schuld for her valuable comments which led to a more solid presentation of the results and S. Yerbí and A. Pérez-Salinas for their worthy insights during QTML 2023. The authors acknowledge financial support from OpenSuperQ+100 (Grant No. 101113946) of the EU

Flagship on Quantum Technologies, as well as from the EU FET-Open project EPIQUS (Grant No. 899368), also from Project Grant No. PID2021-125823NA-I00 595 and Spanish Ramón y Cajal Grant No. RYC-2020-030503-I funded by MCIN/AEI/10.13039/501100011033 and by “ERDF A way of making Europe” and “ERDF Invest in your Future,” this project has also received support from the Spanish Ministry for Digital Transformation and of Civil Service of the Spanish Government through the QUANTUM ENIA project call - Quantum Spain, EU through the Recovery, Transformation and Resilience Plan - NextGenerationEU within the framework of the Digital Spain 2026, and by the EU through the Recovery, Transformation and Resilience Plan - NextGenerationEU within the framework of the Digital Spain 2026 Agenda, we acknowledge funding from the Basque Government through Grant No. IT1470-22 and the IKUR Strategy under the collaboration agreement between Ikerbasque Foundation and BCAM on behalf of the Department of Education of the Basque Government. P.R. and Y.B. acknowledge financial support from the CDTI within the Misiones 2021 program and the Ministry of Science and Innovation under the Recovery, Transformation and Resilience Plan—Next Generation EU under the project “CUCO: Quantum Computing and its Application to Strategic Industries”. Y.B. acknowledges the Ramón y Cajal (RYC2023-042699-I) research fellowship.

APPENDIX A: TRAINING EQKS

One of the main drawbacks of quantum kernel methods is determining the best kernel which depends on the specific problem. In cases where no prior knowledge about the input data is available, approaches have been developed to construct parametrized embedding quantum kernels that are trained for a specific task. These kernel training methods are based on multiple kernel learning [41,42] and kernel target alignment [43].

1. Multiple kernel learning

Multiple kernel learning methods involve the creation of a combined kernel

$$k_{\phi,\omega}(\mathbf{x}_i, \mathbf{x}_j) = f_{\omega}(\{k_{\phi}^{(r)}(\mathbf{x}_i, \mathbf{x}_j)\}_{r=1}^R), \quad (\text{A1})$$

where $k_{\phi}^{(r)}$ represents a set of parametrized embedding quantum kernels, and f_{ω} is a combination function. Two common choices for this function result in either a linear kernel combination

$$k_{\phi,\omega}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{r=1}^R \omega_r k_{\phi}^{(r)}(\mathbf{x}_i, \mathbf{x}_j), \quad (\text{A2})$$

or a multiplicative kernel combination

$$k_{\phi}(\mathbf{x}_i, \mathbf{x}_j) = \prod_{r=1}^R k_{\phi}^{(r)}(\mathbf{x}_i, \mathbf{x}_j). \quad (\text{A3})$$

When it comes to choosing model parameters, in Ref. [41] they use an empirical risk function to minimize, while in Ref. [42] the authors opt for a convex minimization problem. However, in both constructions a complete kernel matrix is computed at each optimization step.

2. Kernel target alignment

Training kernels using kernel target alignment is based on the similarity measure between two kernel matrices, denoted as K_A and K_B . This similarity measure, known as kernel alignment [72], is defined as

$$KA(K_A, K_B) = \frac{\text{tr}(K_A K_B)}{\sqrt{\text{tr}(K_A^2) \cdot \text{tr}(K_B^2)}} \quad (\text{A4})$$

and corresponds to the cosine of the angle between kernel matrices if we see them as vectors in the space of matrices with the Hilbert-Schmidt inner product. The use of kernel alignment to train kernels is by defining the ideal kernel matrix K^* whose entries are $k_{ij}^* = k^*(\mathbf{x}_i, \mathbf{x}_j) = y_i y_j$. Given a parametrized quantum embedding kernel $k_{\theta}(\mathbf{x}_i, \mathbf{x}_j) = |\langle \phi_{\theta}(\mathbf{x}_i) | \phi_{\theta}(\mathbf{x}_j) \rangle|^2$ whose kernel matrix is named as K_{θ} , the kernel-target alignment is defined as the kernel alignment between K and the ideal kernel, i.e.,

$$TA(K_{\theta}) = KA(K_{\theta}, K^*) = \frac{\sum_{ij} y_i y_j k_{\theta}(\mathbf{x}_i, \mathbf{x}_j)}{M \sqrt{\sum_{ij} k_{\theta}(\mathbf{x}_i, \mathbf{x}_j)^2}}, \quad (\text{A5})$$

where we used that $y_i^2 = 1$ independently from the label and M is the number of training points. Thus, authors in Ref. [43] use this quantity as cost function which is minimized using gradient descent over the θ parameters. This strategy is closely related to the approach outlined in Ref. [56], where the authors explore the construction of quantum feature maps with the aim of maximizing the separation between different classes in Hilbert space. However, in the case of the kernel target alignment strategy, similar to multiple kernel learning, the construction of the kernel matrix (or at least a subset of it) at each training step is required.

APPENDIX B: DATA REUPLOADING

To construct a data reuploading QNN for binary classification, each label is associated with a unique quantum state, aiming to maximize the separation in the Bloch sphere. For the single qubit quantum classifier, the labels $+1$ and -1 are represented by the computational basis states $|0\rangle$ and $|1\rangle$, respectively. The objective is to appropriately tune the parameters $\{\theta_l\}_{l=1}^L$ that define the state

$$|\phi_{\theta}(\mathbf{x}_i)\rangle = U(\theta_L) U(\mathbf{x}) \dots U(\theta_1) U(\mathbf{x}) |0\rangle, \quad (\text{B1})$$

to rotate the data points of the same class close to their corresponding label state. To achieve this, we use the fidelity cost function

$$f_{\text{cost}} = \frac{1}{M} \sum_{i=1}^M (1 - |\langle \phi_i^j | \phi_{\theta}(\mathbf{x}_i) \rangle|^2), \quad (\text{B2})$$

where $|\phi_i^j\rangle$ represents the correct label state for the data point $\mathbf{x}_i$. This optimization process occurs on a classical processor. Once the model is trained, the quantum circuit is applied to a test data point $\mathbf{x}_t$, and the probability of obtaining one of the label states is measured. If this probability surpasses a certain threshold (set as $1/2$ in this case), the data point is classified into the corresponding label state class. Formally, the decision

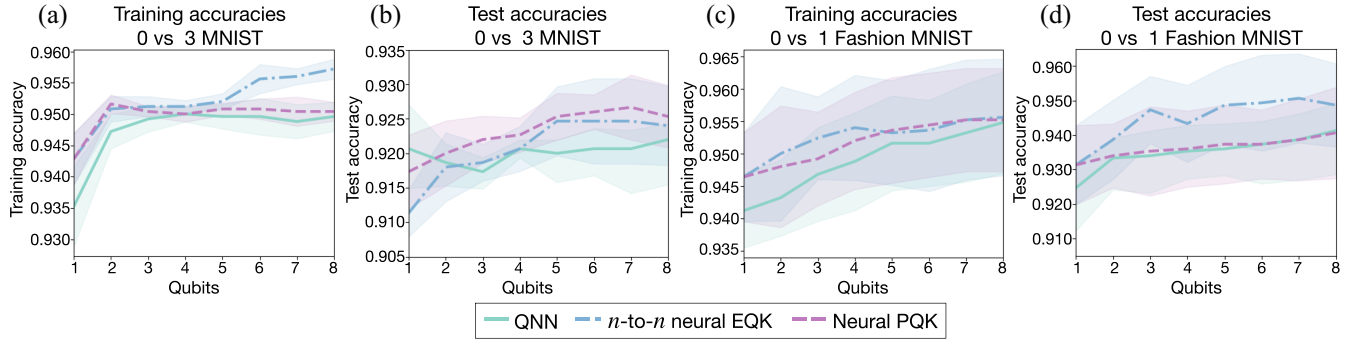


FIG. 11. Mean training [(a) and (c)] and test accuracies [(b) and (d)] using the proposed QNN construction and neural quantum kernels. We present experiments distinguishing between the digits 0 and 3 from the MNIST data set [(a) and (b)] and between t-shirt/top ($\hat{y} = 0$) and trousers ($\hat{y} = 1$) from the Fashion MNIST data set [(c) and (d)]. The results are averaged over 5 runs, using 500 training points and 300 test points in each run.

rule can be expressed as

$$\hat{y}[x_i] = \begin{cases} +1 & \text{if } |\langle 0|\phi(x_i)\rangle|^2 \geq 1/2, \\ -1 & \text{if } |\langle 0|\phi(x_i)\rangle|^2 < 1/2. \end{cases} \quad (\text{B3})$$

When consider a multiqubit architecture, the idea is the same but we need to properly choose the corresponding label states. In this work, for a n -qubit QNN we consider as label states the ones defined by the projectors $|0\rangle\langle 0| \otimes \mathbb{1}^{(n-1)}$ and $|1\rangle\langle 1| \otimes \mathbb{1}^{(n-1)}$, which corresponds to a local measurement in the first qubit.

APPENDIX C: HYPERPLANE DEFINED IN THE BLOCH SPHERE

Here we demonstrate that when the QNN training is perfect, constructing the kernel becomes redundant since the decision measurement remains unchanged.

The objective of the data reuploading QNN is to map all points with label $+1$ to the $|0\rangle$ state on the Bloch sphere, while mapping points with label -1 to the $|1\rangle$ state. In an ideal scenario, all $+1$ points would be rotated to the $|0\rangle$ state and all -1 points to the $|1\rangle$ state. Using this feature map to construct the EQK, all points would be support vectors associated to the same Lagrange multiplier α . The SVM generates a separating hyperplane defined by the equation

$$\sum_{i \in \text{SV}} \alpha_i y_i k(x_i, x) + b = 0. \quad (\text{C1})$$

In this scenario, assuming a balanced data set translates to an equal number of support vectors, denoted as N_{SV} , for each class, and results in $b = 0$ due to symmetry. Working with this equation by considering the sum of support vectors separately for the $+1$ class (SV_{+1}) and -1 class (SV_{-1}), we obtain

$$\begin{aligned} \sum_{i \in \text{SV}_{+1}} \alpha_i y_i |\langle \phi(x_i) | \phi(x) \rangle|^2 + \sum_{i \in \text{SV}_{-1}} \alpha_i y_i |\langle \phi(x_i) | \phi(x) \rangle|^2 \\ = N_{\text{SV}} \alpha |\langle 0 | \phi(x) \rangle|^2 - N_{\text{SV}} \alpha |\langle 1 | \phi(x) \rangle|^2 = 0, \end{aligned} \quad (\text{C2})$$

which is equivalent to the hyperplane defined by

$$|\langle 0 | \phi(x) \rangle|^2 = |\langle 1 | \phi(x) \rangle|^2, \quad (\text{C3})$$

mirroring the decision boundary of the QNN part. Therefore, in the case of a perfect training, the SVM becomes redundant.

Even if the points are not perfectly mapped to their label states and the separating hyperplane of the SVM is not exactly the one from Eq. (C3), as long as all points are correctly classified by the QNN, the SVM will yield the same results. Thus, the SVM part is only meaningful for the single-qubit QNN to construct a single-qubit EQK case when the QNN training is suboptimal and requires us to adjust the measurement of the decision boundary.

APPENDIX D: ADDITIONAL NUMERICAL SIMULATIONS ON REAL DATA SETS

We present additional numerical experiments on two different data sets, corresponding to ideal simulations using our proposed QNN, and two types of neural quantum kernels: the n -to- n neural EQK and the neural PQQ, as discussed in the main text. The results are shown in Fig. 11, displaying both the mean and standard deviation of training and test accuracies over five different runs, using 500 training samples and 300 test samples. The two classification tasks considered are distinguishing between the digits 0 and 3 from the MNIST data set [shown in Fig. 11(a) and 11(b)], and distinguishing between t-shirt/top and trousers from the Fashion MNIST data set [shown in Figs. 11(c) and 11(d)]. The conclusions drawn from these results align with those in the main text: adding qubits improves the performance, and kernel methods constructed from the QNN consistently outperform other approaches. Which of the two proposed neural quantum kernels performs better depends on the classification task at hand. While the neural PQQ achieves better test accuracy for the MNIST data set, the neural EQK performs better for the Fashion MNIST data set.

APPENDIX E: ADDITIONAL NUMERICAL SIMULATIONS ON TOY DATA SETS

1. Data sets

For additional numerical simulations we utilized four artificial data sets depicted in Fig. 12 for evaluating the performance of the proposed neural EQKs. Here we explain how they were created:

Sinus Data set: The sinus data set is defined by a sinusoidal function, specifically $f(x_1) = -0.8 \sin(\pi x_1)$.

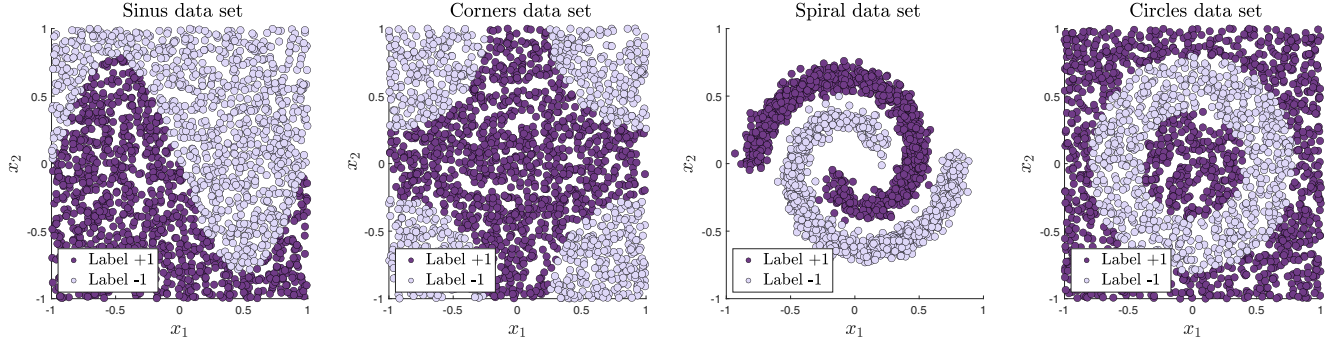


FIG. 12. From left to right: sinus, corners, spiral and circles data sets generated and used in this work. These figures are generated with 2000 data points.

Points located above this sinusoidal curve are categorized as class -1 , while points below it are assigned to class $+1$.

Corners Data set: The corners Data set comprises four quarters of a circumference with a radius of 0.75 , positioned at the corners of a square. Points located inside these circular regions are labeled as class -1 , while points outside are classified as class $+1$.

Spiral Data set: The spiral Data set features two spirals formed by points arranged along a trajectory defined by polar coordinates. The first spiral, denoted as class $+1$, originates at the origin $(0, 0)$ and spirals outward in a counterclockwise direction, forming a curve. The second spiral, labeled class -1 , mirrors the first spiral but spirals inward in a clockwise direction. These spirals are generated by varying the polar angle, selected randomly to create the data points. The radial distance from the origin for each point depends on the angle, creating the characteristic spiral shape. Noise is added to the data points by introducing random perturbations to ensure they do not align perfectly along the spirals.

Circles Data set: The circles Data set is created using two concentric circles that define an annular region. The inner circle has a radius of $\sqrt{2/\pi}$, while the outer circle has a radius of $0.5\sqrt{2/\pi}$. Data points located within the annular region are labeled as -1 , while those outside the region are labeled as $+1$.

2. Additional noisy simulations

We have introduced a combined protocol for binary classification. It is worth noting that the kernel estimation part involves the utilization of a quantum circuit with twice the depth of the QNN part. In practical implementations on current noisy intermediate-scale quantum devices, the consideration of a larger circuit warrants careful attention. This is due to the susceptibility of larger circuits to elevated noise levels, potentially affecting the overall performance.

As previously mentioned, increasing the number of layers in the model enhances its expressivity. However, when constructing the kernel using a high number of layers, noise can have a detrimental impact on the performance, resulting in poorer results for the combined protocol compared to using only the QNN. Therefore, in this section, our objective is to perform simulations to visualize the trade-off between the number of layers in the QNN and using the combined protocol.

To characterize the noise incorporated into these simulations, we employ the operator sum representation of the noise channel ε acting on a quantum state ρ ,

$$\varepsilon(\rho) = \sum_i K_i \rho K_i^\dagger, \quad (\text{E1})$$

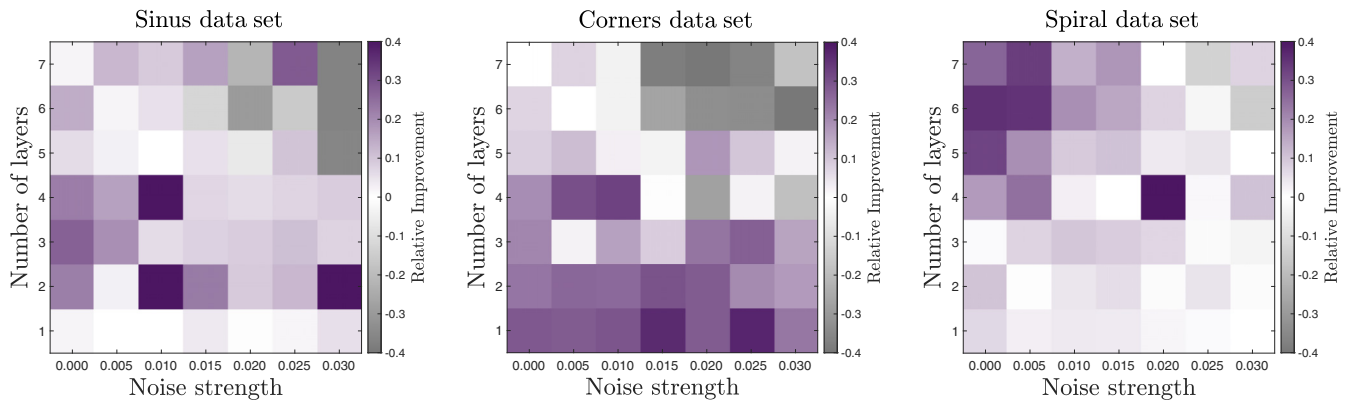


FIG. 13. Relative improvement, as a function of the number of layers and noise strength, comparing the single qubit QNN accuracy performance with the combined protocol utilizing a two-qubit embedding kernel. Results are shown for the sinus, corners, and spiral data sets.

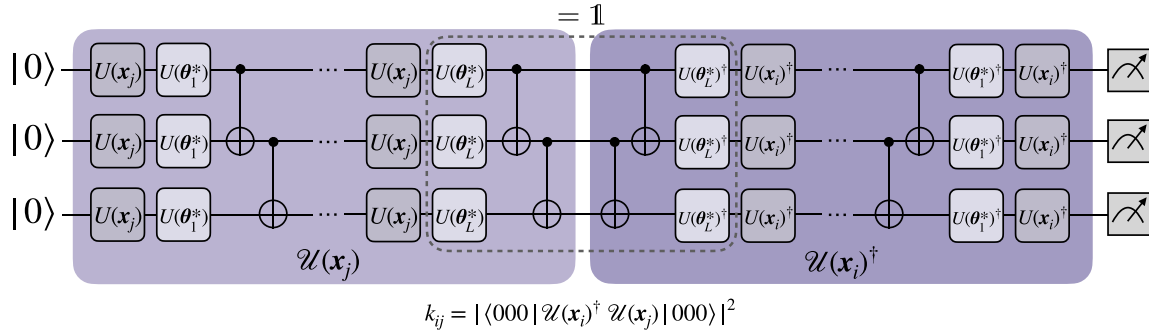


FIG. 14. Explicit quantum circuit for constructing the 1-to-3 EQK with cascade of CNOT gates as source of entanglement. The parameters θ^* are the ones obtained during the training of the single-qubit QNN. Note that the parameters of the last layer are irrelevant.

where K_i represents the respective Kraus operators. In our simulations, we account for two single-qubit noise channels that are applied after each quantum gate. Firstly, we consider amplitude damping error which is described by the Kraus operators

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix}, \quad (\text{E2})$$

$$K_1 = \begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix}. \quad (\text{E3})$$

Here, $\gamma \in [0, 1]$ is the amplitude damping probability, which can be defined as $\gamma = 1 - e^{-\Delta t/T_1}$, with T_1 representing the thermal relaxation time and Δt denoting the duration of application of a quantum gate. The second noise source under consideration is the phase flip error, which can be described by the following Kraus operators:

$$K_0 = \sqrt{1-\alpha} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad (\text{E4})$$

$$K_1 = \sqrt{\alpha} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (\text{E5})$$

In this case, $\alpha \in [0, 1]$ represents the probability of a phase flip error, which can be defined as $2\alpha = 1 - e^{-\Delta t/(2T_2)}$, with T_2 denoting the spin-spin relaxation time or dephasing time. The value of α also falls within the interval $[0, 1]$.

The current state-of-the-art experimental values for the noise parameters of a superconducting quantum processor are as follows: T_1 falls within the range of 50 – 150 μs , T_2 is in the range of 25 – 75 μs , and Δt varies from 10 – 50 ns. It is

TABLE I. Numerical results for the 1-to-3 architecture, considering two types of entangling gates (CNOT and CZ), on four distinct data sets.

Data set	QNN accuracy	EQK type	EQK accuracy
Sinus	0.890	3q CNOT	0.960
Sinus		3q CZ	0.948
Corners	0.886	3q CNOT	0.954
Corners		3q CZ	0.940
Spiral	0.800	3q CNOT	0.994
Spiral		3q CZ	0.866
Circles	0.698	3q CNOT	0.866
Circles		3q CZ	0.864

important to note that noise becomes more pronounced when T_1 and T_2 decrease and/or when Δt increases. To explore the scenario with the worst noise, we consider the extreme values within these ranges, leading to $\gamma^* = 0.001$ and $\alpha^* = 0.0005$.

In our simulations, we simplify the noise characterization by defining a noise strength parameter $\tau \equiv \alpha = \gamma$ for the sake of simplicity. We choose the single-qubit data reuploading the QNN as the kernel selection part. To observe a transition where it is no longer advantageous to include the SVM part, we explore values of τ ranging from 0.005 to 0.030 in steps of 0.005, while also including the noiseless case where $\tau = 0$. These values correspond to examining the range $5\gamma^* \leq \gamma \leq 30\gamma^*$ and $10\alpha^* \leq \alpha \leq 60\alpha^*$. Notably, even in these highly adverse conditions, which are significantly worse than those of current real hardware quantum devices, we find that the combined protocol remains suitable. This conclusion aligns with expectations since the protocol utilizes only a small number of qubits and quantum gates. In Fig. 13, we present the relative improvement, denoted as

$$\text{Relative improvement} = \frac{\text{Acc}_{\text{QNN+SVM}} - \text{Acc}_{\text{QNN}}}{\text{Acc}_{\text{QNN}}}, \quad (\text{E6})$$

plotted as a function of the noise strength parameter τ and the number of layers L . The architecture employed corresponds to the 1-to-2 case, utilizing CNOT gates for entanglement between layers. For QNN training in this instance, we limit it to two epochs with a learning rate of 0.05. The objective is to demonstrate that the resulting EQK is not highly dependent on the training specifics of the corresponding QNN and even a suboptimal QNN training can lead to a powerful EQK for the specific task.

The relative improvements vary depending on the data set under consideration and range from –40% to 40%. As expected, the worst performance of the combined protocol is observed at higher noise levels and for a greater number of layers, corresponding to a longer-depth quantum circuit. It is

TABLE II. Numerical results for the 1-to- n architecture for up to $n = 10$ qubits.

Data set	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$
Corners	0.890	0.954	0.974	0.978	0.982	0.980	0.978	0.978	0.978
Circles	0.832	0.866	0.950	0.970	0.974	0.970	0.978	0.966	0.962

TABLE III. Numerical results for the n -to- n architecture for up to $n = 8$ qubits. These results are depicted in the main text.

Data set	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$
Corners (QNN)	0.886	0.934	0.948	0.952	0.954	0.956	0.960	0.962
Corners (EQK)	0.898	0.948	0.960	0.966	0.970	0.972	0.974	0.974
Circles (QNN)	0.698	0.792	0.820	0.858	0.934	0.944	0.944	0.946
Circles (EQK)	0.796	0.820	0.906	0.968	0.974	0.974	0.976	0.980

crucial to emphasize once more that these high noise strength values significantly surpass the noise parameters of current quantum devices. Therefore, even though the combined protocol necessitates running a double-depth circuit with more qubits, considering the noise model described earlier, it remains suitable for actual quantum devices.

3. Additional ideal simulations

In the main text we provide numerical results for the Fashion MNIST data set. Here we provide the tables of more numerical experiments for the neural EQKs construction using the same hyperparameters: learning rate of 0.05 and 30 epochs for the $n = 1$ QNN and learning rate of 0.005 and 10 epochs for $n > 1$. Figure 14 illustrates the explicit construction of the 1-to-3 EQK using CNOT gates as the source of entanglement. Here, we present additional results for different toy data sets and considering also CZ as source of entanglement. All accuracies refer to test accuracies.

In Table I, the outcomes demonstrate striking similarity when introducing entanglement through CZ gates or CNOT gates, with the exception of the spiral data set. Given the ambiguity surrounding the method of entanglement introduction, one might contemplate utilizing controlled rotations with random parameters as a potential source of entanglement. Notably, it is observed that even starting from a single-qubit QNN achieving accuracies of less than 90%, the construction of EQKs yields accuracies surpassing 95%.

In the 1-to- n construction presented in Table II for the corners and circles data sets, we observe that the accuracy

TABLE IV. Numerical results for the 2-to-2 architecture for different number of layers. For the experiments in the main text we choose $L = 7$.

Data set	Layers	QNN accuracy	EQK accuracy
Sinus	$L = 5$	0.948	0.970
Sinus	$L = 6$	0.956	0.964
Sinus	$L = 7$	0.966	0.972
Sinus	$L = 8$	0.958	0.964
Corners	$L = 5$	0.948	0.970
Corners	$L = 6$	0.936	0.950
Corners	$L = 7$	0.934	0.948
Corners	$L = 8$	0.916	0.920
Spiral	$L = 5$	0.952	0.996
Spiral	$L = 6$	0.974	0.998
Spiral	$L = 7$	0.978	1.000
Spiral	$L = 8$	0.980	0.998
Circles	$L = 5$	0.786	0.812
Circles	$L = 6$	0.808	0.814
Circles	$L = 7$	0.792	0.820
Circles	$L = 8$	0.844	0.902

risks rapidly as qubits are added, reaching a peak at $n = 4$. Subsequently, the accuracy plateaus, attaining a maximum at $n = 6$ for the corners data set and $n = 8$ for the circles data set. After reaching these points, the accuracy gradually decreases with additional qubits.

In the n -to- n approach, as presented in Table III, the accuracy consistently increases with the addition of qubits for both the QNN and the EQK. Additionally, as discussed in the main text, the EQK consistently outperforms the corresponding QNN architecture for the same value of n .

Finally, in Table IV, we present results for the four data sets considering different numbers of layers for the n -to- n architecture with $n = 2$. Adding layers increases the expressivity of the QNN but does not guarantee better accuracies, as we can observe. Again, we see that the EQK consistently outperforms the QNN.

-
- [1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature (London)* **549**, 195 (2017).
 - [2] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, *Rev. Mod. Phys.* **91**, 045002 (2019).
 - [3] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: A review of recent progress, *Rep. Prog. Phys.* **81**, 074001 (2018).
 - [4] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica, G. Salton, S. Wang, and F. G. S. L. Brandão, *Quantum Algorithms: A Survey of Applications and End-To-End Complexities* (Cambridge University Press, Cambridge, 2025).
 - [5] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, *Quantum Sci. Technol.* **4**, 043001 (2019).
 - [6] A. Skolik, S. Jerbi, and V. Dunjko, Quantum agents in the gym: A variational quantum algorithm for deep Q-learning, *Quantum* **6**, 720 (2022).
 - [7] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, *Phys. Rev. A* **101**, 032308 (2020).
 - [8] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, *arXiv:1802.06002*.
 - [9] R. Sweke, J.-P. Seifert, D. Hangleiter, and J. Eisert, On the quantum versus classical learnability of discrete distributions, *Quantum* **5**, 417 (2021).

- [10] S. Jerbi, L. M. Trenkwalder, H. Poulsen Nautrup, H. J. Briegel, and V. Dunjko, Quantum enhancements for deep reinforcement learning in large spaces, *PRX Quantum* **2**, 010328 (2021).
- [11] N. Pirnay, R. Sweke, J. Eisert, and J.-P. Seifert, Superpolynomial quantum-classical separation for density modeling, *Phys. Rev. A* **107**, 042416 (2023).
- [12] C. Gyurik and V. Dunjko, On establishing learning separations between classical and quantum machine learning with classical data, [arXiv:2208.06339](https://arxiv.org/abs/2208.06339).
- [13] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, *Nature (London)* **567**, 209 (2019).
- [14] M. Schuld and N. Killoran, Quantum machine learning in feature hilbert spaces, *Phys. Rev. Lett.* **122**, 040504 (2019).
- [15] 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).
- [16] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, *Phys. Rev. A* **103**, 032430 (2021).
- [17] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, *Quantum* **5**, 582 (2021).
- [18] T. Ono, W. Roga, K. Wakui, M. Fujiwara, S. Miki, H. Terai, and M. Takeoka, Demonstration of a bosonic quantum classifier with data reuploading, *Phys. Rev. Lett.* **131**, 013601 (2023).
- [19] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, Power of data in quantum machine learning, *Nat. Commun.* **12**, 2631 (2021).
- [20] E. Peters, J. Caldeira, A. Ho, S. Leichenauer, M. Mohseni, H. Neven, P. Spentzouris, D. Strain, and G. N. Perdue, Machine learning of high dimensional data on a noisy quantum processor, *npj Quantum Inf.* **7**, 161 (2021).
- [21] K. Bartkiewicz, C. Gneiting, A. Černoč, K. Jiráková, K. Lemr, and F. Nori, Experimental kernel-based quantum machine learning in finite feature space, *Sci. Rep.* **10**, 12356 (2020).
- [22] T. Kusumoto, K. Mitarai, K. Fujii, M. Kitagawa, and M. Negoro, Experimental quantum kernel trick with nuclear spins in a solid, *npj Quantum Inf.* **7**, 94 (2021).
- [23] Y. Wu, B. Wu, J. Wang, and X. Yuan, Quantum phase recognition via quantum kernel methods, *Quantum* **7**, 981 (2023).
- [24] O. Kyriienko and E. B. Magnusson, Unsupervised quantum machine learning for fraud detection, [arXiv:2208.01203](https://arxiv.org/abs/2208.01203).
- [25] J. Tomasi, S. Anthoine, and H. Kadri, Benign overfitting with quantum kernels, [arXiv:2503.17020](https://arxiv.org/abs/2503.17020).
- [26] M. Schuld, Supervised quantum machine learning models are kernel methods, [arXiv:2101.11020](https://arxiv.org/abs/2101.11020).
- [27] M. Schuld and F. Petruccione, Quantum models as kernel methods, in *Machine Learning with Quantum Computers* (Springer International Publishing, Cham, 2021), pp. 217–245.
- [28] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, *Nat. Phys.* **17**, 1013 (2021).
- [29] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, Adaptive Computation and Machine Learning (MIT Press, Cambridge, MA, 2002), p. 644.
- [30] S. Jerbi, L. J. Fiderer, H. P. Nautrup, J. M. Kübler, H. J. Briegel, and V. Dunjko, Quantum machine learning beyond kernel methods, *Nat. Commun.* **14**, 517 (2023).
- [31] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, Quantum classifier with tailored quantum kernel, *npj Quantum Inf.* **6**, 41 (2020).
- [32] I. Salmenperä, I. Kuhtarskis, A. M. van de Griend, and J. K. Nurminen, The impact of feature embedding placement in the ansatz of a quantum kernel in QSVMs, in *Proceedings of IEEE International Conference Quantum Computer Engineering (QCE)* (2024).
- [33] N. Shirai, K. Kubo, K. Mitarai, and K. Fujii, Quantum tangent kernel, *Phys. Rev. Res.* **6**, 033179 (2024).
- [34] J. M. Kübler, S. Buchholz, and B. Schölkopf, The inductive bias of quantum kernels, *Adv. Neural Inf. Process. Syst.* **34**, 12661 (2021).
- [35] Y. Wang and L. Cao, Quantum phase transition detection via quantum support vector machine, *Quantum Sci. Technol.* **10**, 015043 (2025).
- [36] V. Belis, K. A. Woźniak, E. Puljak, P. Barkoutsos, G. Dissertori, M. Grossi, M. Pierini, F. Reiter, I. Tavernelli, and S. Vallecorsa, Quantum anomaly detection in the latent space of proton collision events at the LHC, *Commun. Phys.* **7**, 334 (2024).
- [37] J. R. Glick, T. P. Gujarati, A. D. Córcoles, Y. Kim, A. Kandala, J. M. Gambetta, and K. Temme, Covariant quantum kernels for data with group structure, *Nat. Phys.* **20**, 479 (2024).
- [38] J. J. Meyer, M. Mularski, E. Gil-Fuster, A. A. Mele, F. Arzani, A. Wilms, and J. Eisert, Exploiting symmetry in variational quantum machine learning, *PRX Quantum* **4**, 010328 (2023).
- [39] M. Larocca, F. Sauvage, F. M. Sbahi, G. Verdon, P. J. Coles, and M. Cerezo, Group-invariant quantum machine learning, *PRX Quantum* **3**, 030341 (2022).
- [40] S. Thanasilp, S. Wang, M. Cerezo, and Z. Holmes, Exponential concentration in quantum kernel methods, *Nat. Commun.* **15**, 5200 (2024).
- [41] S. S. Vedaie, M. Noori, J. S. Oberoi, B. C. Sanders, and E. Zahedinejad, Quantum multiple kernel learning, [arXiv:2011.09694](https://arxiv.org/abs/2011.09694).
- [42] A. Ghukasyan, J. S. Baker, O. Goktas, J. Carrasquilla, and S. K. Radha, Quantum-classical multiple kernel learning, [arXiv:2305.17707](https://arxiv.org/abs/2305.17707).
- [43] T. Hubregtsen, D. Wierichs, E. Gil-Fuster, P.-J. H. S. Derks, P. K. Faehrmann, and J. J. Meyer, Training quantum embedding kernels on near-term quantum computers, *Phys. Rev. A* **106**, 042431 (2022).
- [44] P. Rodriguez-Grasa, R. Farzan-Rodriguez, G. Novelli, Y. Ban, and M. Sanz, Satellite image classification with neural quantum kernels, *Mach. Learn.: Sci. Technol.* **6**, 015043 (2025).
- [45] J. Bowles, S. Ahmed, and M. Schuld, Better than classical? The subtle art of benchmarking quantum machine learning models, [arXiv:2403.07059](https://arxiv.org/abs/2403.07059).
- [46] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, *Phys. Rev. Lett.* **113**, 130503 (2014).
- [47] S. Park, D. K. Park, and J.-K. K. Rhee, Variational quantum approximate support vector machine with inference transfer, *Sci. Rep.* **13**, 3288 (2023).
- [48] E. Gil-Fuster, J. Eisert, and V. Dunjko, On the expressivity of embedding quantum kernels, *Mach. Learn.: Sci. Technol.* **5**, 025003 (2024).

- [49] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, Quantum fingerprinting, *Phys. Rev. Lett.* **87**, 167902 (2001).
- [50] M. Fanizza, M. Rosati, M. Skotiniotis, J. Calsamiglia, and V. Giovannetti, Beyond the swap test: Optimal estimation of quantum state overlap, *Phys. Rev. Lett.* **124**, 060503 (2020).
- [51] L. Cincio, Y. Subaşı, A. T. Sornborger, and P. J. Coles, Learning the quantum algorithm for state overlap, *New J. Phys.* **20**, 113022 (2018).
- [52] R. Shaydulin and S. M. Wild, Importance of kernel bandwidth in quantum machine learning, *Phys. Rev. A* **106**, 042407 (2022).
- [53] A. Canatar, E. Peters, C. Pehlevan, S. M. Wild, and R. Shaydulin, Bandwidth enables generalization in quantum kernel models, *Trans. Mach. Learn. Res.* (2023).
- [54] Y. Suzuki, H. Kawaguchi, and N. Yamamoto, Quantum Fisher kernel for mitigating the vanishing similarity issue, *Quantum Sci. Technol.* **9**, 035050 (2024).
- [55] F. Sauvage, M. Larocca, P. J. Coles, and M. Cerezo, Building spatial symmetries into parameterized quantum circuits for faster training, *Quantum Sci. Technol.* **9**, 015029 (2024).
- [56] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning, [arXiv:2001.03622](https://arxiv.org/abs/2001.03622).
- [57] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, *Nat. Commun.* **9**, 4812 (2018).
- [58] M. Ragone, B. N. Bakalov, F. Sauvage, A. F. Kemper, C. O. Marrero, M. Larocca, and M. Cerezo, A Lie algebraic theory of barren plateaus for deep parameterized quantum circuits, *Nat. Commun.* **15**, 7172 (2024).
- [59] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, *PRX Quantum* **3**, 010313 (2022).
- [60] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, *Nat. Commun.* **12**, 1791 (2021).
- [61] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, Noise-induced barren plateaus in variational quantum algorithms, *Nat. Commun.* **12**, 6961 (2021).
- [62] S. Thanasilp, S. Wang, N. A. Nghiem, P. Coles, and M. Cerezo, Subtleties in the trainability of quantum machine learning models, *Quantum Mach. Intell.* **5**, 21 (2023).
- [63] G. Li, R. Ye, X. Zhao, and X. Wang, Concentration of data encoding in parameterized quantum circuits, *Adv. Neural Inf. Process. Syst.* **35**, 19456 (2022).
- [64] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, and M. Cerezo, Barren plateaus in variational quantum computing, *Nat. Rev. Phys.* **7**, 174 (2025).
- [65] E. R. Anschuetz, A unified theory of quantum neural network loss landscapes, [arXiv:2408.11901](https://arxiv.org/abs/2408.11901).
- [66] B. Casas and A. Cervera-Lierta, Multidimensional Fourier series with quantum circuits, *Phys. Rev. A* **107**, 062612 (2023).
- [67] A. Barthe and A. Pérez-Salinas, Gradients and frequency profiles of quantum re-uploading models, *Quantum* **8**, 1523 (2024).
- [68] A. Pérez-Salinas, D. López-Núñez, A. García-Sáez, P. Forn-Díaz, and J. I. Latorre, One qubit as a universal approximant, *Phys. Rev. A* **104**, 012405 (2021).
- [69] I. Panadero, Y. Ban, H. Espinós, R. Puebla, J. Casanova, and E. Torrontegui, Regressions on quantum neural networks at maximal expressivity, [arXiv:2311.06090](https://arxiv.org/abs/2311.06090).
- [70] H. Xiao, K. Rasul, and R. Vollgraf, Fashion-MNIST: A novel image data set for benchmarking machine learning algorithms, [arXiv:1708.07747](https://arxiv.org/abs/1708.07747).
- [71] E. Gil-Fuster, J. Eisert, and C. Bravo-Prieto, Understanding quantum machine learning also requires rethinking generalization, *Nat. Commun.* **15**, 2277 (2024).
- [72] D. Holmes and L. Jain, *Innovations in Machine Learning: Theory and Applications* (Springer, New York, 2006).